Certified Tester Advanced Level

Test Management Syllabus

v3.0

International Software Testing Qualifications Board

BSTQB®
CTAL-TM 3.0

Aviso de Copyright

Histórico de Revisões

Versão	Data	Comentários
ISEB v1.1	2001/09/04	ISEB Practitioner Syllabus
ISTQB 1.2E	2003/09	ISTQB Advanced Level Syllabus from EOQ-SG
V2007	2007/10/12	Certified Tester Advanced Level syllabus versão 2007
D100626	2010/06/10	Incorporação das alterações aceitas em 2009, separação de cada capítulo para os módulos separados
D101227	2010/12/10	Aceite de alterações no formato e correções que não tenham impacto sobre o significado das frases.
D2011	2011/10/31	Mudança para dividir o syllabus, OAs reelaborados e mudanças no texto para corresponder aos OAs. Adição de ONs.
Alpha 2012	2012/02/09	Incorporação de todos os comentários dos MBs recebidos da versão de outubro.
Beta 2012	2012/03/26	Incorporação dos comentários dos MBs recebidos a tempo da versão Alpha.
Beta 2012	2012/04/07	Versão beta enviada à GA
Beta 2012	2012/06/08	Versão editada liberada para os MBs
Beta 2012	2012/06/27	Incorporação dos comentários do EWG e do Glossário
RC 2012	2012/08/15	Versão candidata a lançamento - edições finais do MB incluídas
Beta 3.0	2023/10/31	Incorporação de todos os comentários dos conselhos de administração recebidos para todas as seções (BOIncs) da revisão Alfa
POST Beta 3.0	2024/01/31	Incorporação de todos os comentários dos conselhos de membros recebidos para todas as seções (BOIncs) da revisão Beta
POST Beta 3.0	2024/02/29	Pequenas modificações na revisão
RC v3.0	2024/03/28	Versão candidata a lançamento - últimas alterações formais no modelo incluídas conforme proposto pelo PWG
V.3.0	2024/05/03	Retrabalho após o lançamento; correção de erros de digitação e inconsistências eliminadas

Histórico de Revisões na Língua Portuguesa

Versão	Data	Comentários
RC	04/06/2024	Release Candidate da versão traduzida do syllabus
1	08/07/2024	Versão de lançamento na língua portuguesa
2	06/01/2025	Adequação do documento ao novo layout do ISTQB

Índice

Reconhecimento

Este documento foi formalmente lançado pela Assembleia Geral do ISTQB® em 3 de maio de 2024 e foi produzido por uma equipe do International Software Testing Qualifications Board formada por: Horst Pohlmann (Product owner, vice-presidente do AELWG), Tauhida Parveen, Francis Fenner, Laura Albert, Matthias Hamburg, Maud Schlich, Tanja Tremmel, Ralf Bongard, Erik van Veenendaal, Jan Giessen, Bernd Freimut, Andreas Neumeister, Georg Sehl, Rabi Arabi, Therese Kuhfuß, Ecaterina Irina Manole, Veronica Belcher, Kenji Onishi, Pushparajan Balasubramanian, Meile Postuma e Miroslav Renda.

A equipe agradece a Gary Mogyorodi por sua revisão técnica (em Beta), a Julia Sabatine por sua revisão, à equipe de revisão e aos Conselhos de Membros por suas sugestões e contribuições. As seguintes pessoas participaram da revisão, dos comentários e da votação deste syllabus:

Revisão Alpha: Benjamin Timmermans, Mattijs Kemmink, Rik Marselis, Jean-Francois Riverin, Gary Mogyorodi, Ralf Bongard, Ingvar Nordström, Yaron Tsubery, Imre Mészáros, Mattijs Kemmink, Ádám Bíró, Ramit M Kaul, Chinthaka Indikadahena, Darvay Tamás Béla, Beata Karpinska, Young jae Choi, Stuart Reid, Tal Pe'er, Meile Posthuma, Daniel van der Zwan, Klaudia Dussa-Zieger, Jörn Münzel, Ralf Bongard, Petr Neugebauer, Derk-Jan de Grood, Rik Kochuyt, Andreas Hetz, Laura Albert, Eszter Sebestyeni, Tamás Szőke, Henriett Braunné Bokor, Ágota Horváth, Péter Sótér, Ferenc Hamori, Darvay Tamás Béla, Paul Weymouth, Lloyd Roden, Kevin Chen, Huang qin, Pushparajan Balasubramanian, Szilard Szell, Tamas Stöckert, Lucjan Stapp, Adam Roman, Anna Miazek, Márton Siska, Erhardt Wunderlich, László Kvintovics, Murian Song, Mette Bruhn-Pedersen, Petra Schneider, Michael Stahl, Ramit M Kaul, Imre Mészáros, Dilhan Jayakody, Francisca Cano Ortiz, Johan Klintin, Liang Ren, Ole Chr. Hansen, Zsolt Hargitai, Tamás Rakamazi, Kenji Onishi, Arnika Hryzszko, Rabih Arabi, Veronica Belcher, and Vignesh Balasubramanian.

Revisão Beta: Maria-Therese Teichmann, Dominik Weber, Thomas Puffler, Peter Kunit, Martin Klonk, Michaël Pilaeten, Wim Decoutere, Arda Ender Torçuk, Piet de Roo, Rik Marselis, Jakub Platek, Ding Guofu, Zheng Dandan, Liang Ren, Yifan Chen, Hallur Helmsdal, Ole Chr. Hansen, Klaus Skafte, Gitte Ottosen, Tanzeela Gulzar, Arne Becher, Klaudia Dussa-Zieger, Jan Giesen, Florian Fieber, Carsten Weise, Arnd Pehl, Matthias Hamburg, Stephanie Ulrich, Jürgen Beniermann, Márton Siska, Sterbinszky Ádám, Ágnes Srancsik, Marton Matyas, Tamas Stöckert, Csilla Varga, Zsolt Hargitai, Bíró Ádám, Horváth Ágota, Sebestyéni Eszter, Szilárd Széll, Péter Sótér, Giancarlo Tomasig, Nicola de Rosa, Kaiwalya Katyarmal, Pradeep Tiwari, Sreeja Padmakumari, Seunghee Choi, Stuart Reid, Dmitrij Nikolajev, Mantas Aniulis, Monika Stoecklein-Olsen, Adam Roman, Mahmoud Khalaili, Ingvar Nordström, Beata Karpinska, Armin Born, Ferdinand Gramsamer, Mergole Kuaté, Thomas Letzkus, Nishan Portoyan, Ainsley Rood, Lloyd Roden, Sarah Ireton.

O syllabus do Advanced Test Manager 2010-2012 foi produzido por uma equipe do subgrupo de trabalho de nível avançado do International Software Testing Qualifications Board - Advanced Test Manager: Rex Black (presidente), Judy McKay (vice-presidente), Graham Bath, Debra Friedenberg, Bernard Homès, Paul Jorgensen, Kenji Onishi, Mike Smith, Geoff Thompson, Erik van Veenendaal, Tsuyoshi Yumoto. A equipe principal agradece à equipe de revisão e aos Conselhos Nacionais por suas sugestões e contribuições. Quando o Syllabus de Nível Avançado foi concluído, o Advanced Level Working Group tinha os seguintes membros (ordem alfabética):Graham Bath, Rex Black, Maria Clara Choucair, Debra Friedenberg, Bernard Homès (Vice Chair), Paul Jorgensen, Judy McKay, Jamie Mitchell, Thomas Mueller, Klaus Olsen, Kenji Onishi, Meile Posthuma, Eric Riou du Cosquer, Jan Sabak, Hans Schaefer, Mike Smith (Chair), Geoff Thompson, Erik van Veenendaal, Tsuyoshi Yumoto.

As seguintes pessoas participaram da revisão, dos comentários e da votação deste syllabus:

Chris van Bael, Graham Bath, Kimmo Hakala, Rob Hendriks, Marcel Kwakernaak, Rik Marselis, Don Mills, Gary Mogyorodi, Thomas Mueller, Ingvar Nordstrom, Katja Piroué, Miele Posthuma, Nathalie Rooseboom de Vries, Geoff Thompson, Jamil Wahbeh, Hans Weiberg.

O BSTQB® agradece à equipe do Grupo de Traduções do BSTQB pelo empenho em traduzir este material. Atuaram na tradução e revisão: Eduardo Rodrigues Medeiros, George Fialkovitz, Irene Nagase, Osmar Higashi, Paula Oliveira, Rogério Athaide de Almeida, Stênio Viveiros.

O BSTQB® também agradece aos ISTQB® Accredited Training Providers no Brasil que contribuíram na revisão da tradução com sugestões e questionamentos. No momento deste trabalho os seguintes provedores estavam credenciados: Conecteseaqui, Exseed, Iterasys, Keeggo.

O BSTQB® também agradece às empresas brasileiras credenciadas no ISTQB® Partner Program que contribuíram na revisão da tradução com sugestões e questionamentos. No momento deste trabalho as seguintes empresas estavam credenciadas: Deal, Deltapoint, Itriad, Keeggo, Venturus.

0 Introdução

0.1 Objetivo deste site Syllabus

Esse syllabus forma a base para a Qualificação Internacional em Teste de Software no Nível Avançado para Gerenciamento de Testes. O ISTQB® fornece esse syllabus da seguinte forma:

- Aos Conselhos Membros, para traduzir para seu idioma local e credenciar os provedores de treinamento. Os Conselhos Membros podem adaptar o syllabus às suas necessidades específicas de idioma e modificar as referências para adaptá-las às suas publicações locais.
- Aos órgãos de certificação, para que elaborem questões de exame em seu idioma local, adaptadas aos objetivos de aprendizagem deste programa.
- Para os provedores de treinamento credenciados pelo ISTQB® , para produzir material didático e determinar os métodos de ensino apropriados.
- Para os candidatos à certificação, para se prepararem para o exame de certificação (o ISTQB® recomenda fazer um treinamento credenciado pelo ISTQB® antes de participar de um exame de nível avançado).
- Para a comunidade internacional de Engenharia de Software e Sistemas, para promover a profissão de teste de software e sistemas, e como base para livros e artigos.

0.2 O Certified Tester Advanced Level, Test Management

Essa certificação de nível avançado destina-se a qualquer pessoa envolvida no gerenciamento de testes de software. Isso inclui pessoas em funções como testadores, consultores de teste, gerentes de teste, testadores de aceite do usuário, scrum masters, gerentes de projeto ou proprietários de produtos. Essa certificação de nível avançado de gerenciamento de teste também é adequada para qualquer pessoa que queira ter um entendimento avançado de testes de software, como gerentes de projeto, gerentes de qualidade, gerentes de desenvolvimento de software, analistas de negócios, diretores de TI e consultores de gerenciamento. Os detentores do Certified Tester Advanced Level, Test Management poderão obter as qualificações de teste de software de Nível Especialista do ISTQB®. O certificado ISTQB® Certified Tester Advanced Level, Test Manager ou Test Management é válido por toda a vida e não precisa ser renovado. O certificado é reconhecido internacionalmente e demonstra a competência profissional e a credibilidade dos candidatos no gerenciamento de testes.

0.3 Carreira para testadores

O esquema do ISTQB® oferece suporte para profissionais de teste em todos os estágios de suas carreiras. As pessoas que obtiverem a certificação ISTQB® Certified Tester Advanced Level Test Management também podem se interessar pelas outras certificações Core Advanced Level (ou seja, Test Analyst e Technical Test Analyst) e, posteriormente, pelas certificações ISTQB® Expert Level (ou seja, Test Management ou Improving the Test Process). Qualquer pessoa que queira desenvolver habilidades de teste no desenvolvimento de software ágil pode considerar as certificações Agile Technical Tester ou Agile Test Leadership at Scale. O fluxo de Especializações oferece um mergulho profundo em áreas que têm abordagens de teste e atividades de teste específicas (p. ex., em automação de teste, teste de IA ou teste de aplicativo móvel), ou conhecimento de teste de cluster para determinados domínios do setor (p. ex., automotivo ou jogos). Acesse www.istqb.org para obter as informações mais recentes sobre o Esquema de Certificação em Testes do ISTQB®.

0.4 Resultados de negócio

Esta seção lista os onze resultados comerciais esperados de um candidato que tenha obtido a Certified Tester Advanced Level, Test Management.

Um testador certificado em gerenciamento de testes de nível avançado pode...

TM_01	Gerenciar testes em vários projetos de desenvolvimento de software aplicar processos de gerenciamento de testes estabelecidos para a equipe do projeto ou organização de testes
TM_02	Identificar os stakeholders no teste e os modelos de ciclo de vida de desenvolvimento de software que são relevantes em um determinado contexto
TM_03	Organizar a identificação de riscos e avaliação de riscos em qualquer ciclo de vida de desenvolvimento de software e use os resultados para orientar os testes a fim de atingir os objetivos do teste
TM_04	Definir uma estratégia de teste do projeto consistente com a estratégia de teste organizacional e o contexto do projeto
TM_05	Monitorar e controlar continuamente os testes para atingir as metas do projeto
TM_06	Avaliar e relatar o progresso do teste aos stakeholders do projeto
TM_07	Identificar as habilidades necessárias e desenvolvê-las em sua equipe
TM_08	Preparar e apresentar um caso de negócios para testes em diferentes contextos que descreva os custos e os benefícios esperados
TM_09	Liderar atividades de aprimoramento do processo de teste Atividades de aprimoramento do processo de teste em projetos ou fluxos de produtos de desenvolvimento de software e contribuição para iniciativas organizacionais de aprimoramento do processo de teste
TM_10	Planejar as atividades de teste, incluindo a infraestrutura de teste necessária, e estimar o esforço necessário para testar
TM_11	Criar relatórios de defeitos e um fluxo de trabalho de defeitos adequado para um ciclo de vida de desenvolvimento de software

0.5 Objetivos de Aprendizagem e Níveis Cognitivos de Conhecimento

Os objetivos de aprendizagem apoiam os resultados comerciais e são usados para criar os exames do Certified Tester Advanced Test Management.

Em geral, todo o conteúdo deste syllabus pode ser examinado em nível K1, exceto a Introdução, as Referências, o Epílogo e os Apêndices. Ou seja, o candidato pode ser solicitado a reconhecer, lembrar ou recordar uma palavra-chave ou conceito mencionado em qualquer um dos três capítulos deste syllabus. Os objetivos específicos de aprendizado e os níveis correspondentes são mostrados no início de cada capítulo e classificados da seguinte forma:

- K2: Compreender
- K3: Aplicar
- K4: Analisar

Mais detalhes e exemplos de objetivos de aprendizado são fornecidos no Apêndice A.

Todos os termos listados como palavras-chave logo abaixo dos títulos dos capítulos devem ser lembrados (K1), mesmo que não sejam explicitamente mencionados nos objetivos de aprendizado.

0.6 Exame de certificação Advanced Level Test Management

O exame de certificação do *Certified Tester Advanced Test Management* baseia-se neste syllabus. As respostas às questões do exame podem exigir o uso de material baseado em mais de uma seção deste syllabus. Todas as seções do syllabus são passíveis de exame, exceto a Introdução, os Apêndices e as Referências. Normas e livros são

incluídos como referências (Capítulo 5), mas seu conteúdo, além do que está resumido no próprio syllabus a partir dessas normas e livros, não é passível de exame

Consulte o documento "*Exam Structures and Rules 1.1 Compatible with Syllabus Foundation and Advanced Levels and Specialists Modules*" para obter mais detalhes.

O critério de entrada para fazer o exame *Certified Tester Advanced Test Management* é:

- Os candidatos devem ter o certificado *ISTQB® Foundation Level* antes de fazer o exame de certificação *Certified Tester Advanced Test Management*. No entanto, é altamente recomendável que o candidato tenha pelo menos um histórico mínimo em desenvolvimento ou teste de software, como, por exemplo, dois anos de experiência como testador ou desenvolvedor de software.

0.7 Credenciamento

Um Conselho de Membros do ISTQB® pode credenciar provedores de treinamento cujo material de curso siga este syllabus. Os provedores de treinamento devem obter as diretrizes de credenciamento do Conselho de Membros ou do órgão que realiza o credenciamento. Um curso credenciado é reconhecido como estando em conformidade com este syllabus e pode ter um exame ISTQB® como parte do curso.

As diretrizes de credenciamento para este syllabus são as Diretrizes de Credenciamento genéricas publicadas pelo Grupo de Trabalho de Gerenciamento de Processos e Conformidade do ISTQB® .

0.8 Manuseio de padrões

Há padrões referenciados no syllabus *Certified Tester Advanced Level, Test Management* (p. ex., ISO, IEC e IEEE). O objetivo dessas referências é fornecer uma estrutura ou uma fonte de informações adicionais, se o leitor desejar. Observe que o syllabus usa esses padrões como referências. As normas não se destinam a exame. Consulte o capítulo 5, *Referências*, para obter mais informações sobre normas.

0.9 Nível de detalhe

O nível de detalhamento desse syllabus permite cursos e exames consistentes em nível internacional. Para atingir esse objetivo, o syllabus consiste em:

- Objetivos gerais de instrução que descrevem a intenção do syllabus de gerenciamento de testes de nível avançado
- Uma lista de palavras-chave que os alunos devem ser capazes de lembrar
- Objetivos de aprendizado para cada área de conhecimento, descrevendo o resultado cognitivo de aprendizado a ser alcançado
- Uma descrição dos principais conceitos, incluindo referências a fontes, como literatura ou padrões aceitos

O conteúdo do syllabus não é uma descrição de toda a área de conhecimento de testes de software; ele reflete o nível de detalhes a ser abordado nos cursos de treinamento de gerenciamento de testes de nível avançado. Ele se concentra em conceitos e técnicas de teste que podem ser aplicados a todos os projetos de software.

0.10 Como este syllabus está organizado

Há três capítulos com conteúdo passível de exame. O cabeçalho de nível superior de cada capítulo especifica o tempo mínimo necessário para que os cursos de treinamento credenciados cubram o conteúdo do capítulo; o tempo não é fornecido abaixo do nível do capítulo. Para os cursos de treinamento credenciados, o syllabus exige um mínimo de 22,75 horas de instrução, distribuídas nos três capítulos da seguinte forma:

Capítulo 1: Gerenciando as atividades de teste (750 minutos)

- O participante aprende a explicar as atividades de gerenciamento de testes (ou seja, planejamento de testes, monitoramento de testes, controle de testes e conclusão do teste)
- O participante aprende a definir uma estratégia de teste de projeto incluindo os objetivos do teste e selecionando a abordagem de teste adequada e consistente com a estratégia de teste organizacional e o contexto do projeto
- O participante aprende a gerenciar projetos em vários contextos.
- O participante aprende a aplicar testes baseados em riscos a concentrar os testes nos riscos identificados
- O participante aprende a liderar as atividades de aprimoramento do processo de teste realizando uma retrospectiva de projeto ou iteração.
- O participante aprende a melhorar o suporte de ferramentas para testes, levando em consideração os riscos, os custos e os benefícios do suporte de ferramentas.

Capítulo 2: Gerenciando o produto (390 minutos)

- O participante aprende a monitorar e controlar os testes para atender aos objetivos do teste usando métricas de teste e a avaliar e relatar o progresso do teste.
- O participante aprende a selecionar técnicas adequadas de estimativa de teste em diferentes subsidiárias, seguindo diferentes modelos de desenvolvimento de software.
- O participante aprende a definir um fluxo de trabalho de defeitos no gerenciamento de defeitos para se adequar ao desenvolvimento de software sequencial, Ágil e híbrido.

Capítulo 3: Gerenciando a equipe (225 minutos)

- O participante aprende a analisar um determinado contexto de projeto para identificar as habilidades necessárias para a equipe de teste
- O participante aprende a gerenciar uma equipe de acordo com a abordagem de equipe completa
- O participante aprende a definir um caso de negócios para as atividades de teste no projeto

Nota: Para cada objetivo de aprendizado, existe no syllabus atual uma subseção correspondente com o conteúdo (p. ex., para o OA-1.2.3, existe a subseção 1.2.3).

0.11 Quais são as premissas fundamentais deste syllabus?

Este syllabus destina-se a qualquer pessoa que queira atingir um nível avançado de competência em gerenciamento de testes, como gerentes de testes, analistas de testes, engenheiros de testes, consultores de testes, coordenadores de testes, líderes de testes e gerentes de projetos. O syllabus está alinhado com o *ISTQB® Foundation Level Syllabus v4*, que fornece o conhecimento básico e a compreensão dos testes de software.

Este syllabus abrange duas funções principais nos testes: a função de gerenciamento de testes e a função de testes. A função de gerenciamento de testes também é conhecida como Gerente de Teste no contexto do modelo de desenvolvimento sequencial, em que o Gerente de Teste é normalmente uma função separada do gerente de projeto ou do Product Owner. A função de gerenciamento de testes é responsável pelo processo geral de testes, pela equipe de testes e pelo gerenciamento de testes. Isso inclui definir a estratégia de teste, planejar as atividades de teste, monitorar e controlar o progresso do teste, relatar os resultados do teste e gerenciar os riscos e problemas do teste. A função de gerenciamento de testes também garante que os objetivos do teste estejam alinhados com as necessidades do negócio e dos stakeholders e que as atividades de teste sejam coordenadas com outros stakeholders do projeto.

A função de teste também executa a avaliação de testes, o gerenciamento de defeitos e as atividades de encerramento de testes. A função de teste usa várias técnicas de teste para garantir a qualidade e a confiabilidade dos artefatos de teste e do sistema em teste. A função de teste também usa ferramentas de teste e automação para

dar suporte ao processo de teste e melhorar a eficiência e a eficácia do teste. As atividades e tarefas atribuídas a essas funções podem variar de acordo com o contexto, como o projeto, o produto, as habilidades e a organização. (consulte o *ISTQB® Foundation Level Syllabus v4*).

O termo *membro da equipe de teste* é usado neste syllabus para se referir a qualquer pessoa em uma função de gerenciamento de teste ou teste que realize testes, independentemente do contexto organizacional e de outras funções. As equipes de teste são compostas por indivíduos com diferentes habilidades e competências. Os membros da equipe também podem ter níveis variados de experiência e certificação, como nível básico, avançado ou especializado. Os membros da equipe de teste também podem ter funções e responsabilidades diferentes, dependendo da abordagem de teste e do modelo de processo de teste usado, como teste Ágil, teste baseado em modelo, teste baseado em risco etc.

Um ponto importante sobre a perspectiva que o syllabus oferece é o fato de que ele se concentra no gerenciamento de testes no nível do projeto e não no gerenciamento de testes no nível organizacional. Portanto, esse syllabus está em conformidade e contém informações que podem ser usadas para atividades de gerenciamento de testes em nível de projeto, mas não tanto para o gerenciamento de testes em nível organizacional.

O desenvolvimento híbrido de software é usado neste syllabus para se referir a uma abordagem de desenvolvimento de software que combina elementos de diferentes modelos de ciclo de vida de software, como o modelo V, iterativo, incremental e ágil. O desenvolvimento híbrido de software tem como objetivo aproveitar os pontos fortes e atenuar os pontos fracos de cada modelo, dependendo do contexto e das necessidades do projeto. Por exemplo, uma abordagem de desenvolvimento de software híbrido pode usar um modelo em V para as fases iniciais de planejamento e análise de requisitos, seguido por um modelo ágil para as fases de projeto, desenvolvimento e teste. Como alternativa, uma abordagem de desenvolvimento de software híbrido pode usar um modelo iterativo para o gerenciamento geral do projeto, enquanto aplica um modelo incremental para cada iteração e um modelo Ágil para cada incremento. O desenvolvimento de software híbrido exige muita flexibilidade, comunicação e colaboração entre os stakeholders, bem como uma compreensão clara dos objetivos, riscos e restrições de cada fase e modelo.

De acordo com este syllabus e o Glossário, uma estratégia de teste é uma descrição de como o teste será realizado para atingir os objetivos do teste em determinadas circunstâncias. Uma estratégia de teste define o escopo geral, a abordagem e os recursos para testar um sistema ou um produto. Normalmente, ela é documentada em um plano de teste ou como parte de outros documentos, dependendo do contexto do teste. Uma estratégia de teste é influenciada pela estratégia de teste organizacional, que é uma estratégia de teste de alto nível que descreve como o teste é feito em uma organização. Uma estratégia de teste também pode existir para um único nível de teste ou um tipo de teste, que são aspectos específicos do teste que se concentram em diferentes objetivos, metas e critérios. O termo genérico "estratégia de teste" pode ser usado em qualquer contexto (projeto, organização, produto). Uma abordagem de teste é a maneira pela qual as tarefas de teste são implementadas, especialmente a seleção e a combinação de níveis de teste, tipos de teste e técnicas de teste para testes estáticos e dinâmicos, bem como outras práticas de teste, como testes com script, testes manuais, testes consecutivos etc. A abordagem de teste escolhida pela função de gerenciamento de teste é uma decisão fundamental na formulação de uma estratégia de teste apropriada para um determinado contexto.

1 Gerenciando as Atividades de Teste – 750 min

Palavras-chave

testes baseados em experiência, testes funcionais, modelo de desenvolvimento de software híbrido, modelo de desenvolvimento incremental, modelo de desenvolvimento iterativo, teste não-funcional, risco do produto, risco do produto , risco de qualidade, retrospectiva, análise de risco, avaliação de riscos, identificação de riscos, impacto do risco, nível de risco, probabilidade do risco, gerenciamento de riscos, mitigação de riscos, monitoramento de riscos, teste baseado em risco, modelo de desenvolvimento sequencial, metodologia de metas S.M.A.R.T, ciclo de vida de desenvolvimento de software, conclusão de testes, controle de teste, nível de teste, Test Maturity Model Integrated, monitoramento de teste, objetivo do teste, plano de teste, planejamento de teste, melhoria do processo de teste, estratégia de teste, tipo de teste, TPI NEXT

Palavras-chave específicas do domínio

métrica de meta e questão (GQM), IDEAL, indicador, medida, métrica

Objetivos de aprendizagem:

1.1 O Processo de Teste

TM-1.1.1 (K2) Resumir o planejamento de testes

TM-1.1.2 (K2) Resumir o monitoramento de testes e controle de testes

TM-1.1.3 (K2) Resumir a conclusão do teste

1.2 O Contexto do Teste

TM-1.2.1 (K2) Comparar por que diferentes stakeholders estão interessados em testes

TM-1.2.2 (K2) Explicar por que o conhecimento dos stakeholders é importante no gerenciamento de testes

TM-1.2.3 (K2) Explicar os testes em um modelo de desenvolvimento de software híbrido

TM-1.2.4 (K2) Resumir as atividades de gerenciamento de testes para vários ciclos de vida de desenvolvimento de software

TM-1.2.5 (K2) Comparar as atividades de gerenciamento de testes em vários níveis de teste

TM-1.2.6 (K2) Comparar as atividades de gerenciamento de testes para vários tipos de testes

TM-1.2.7 (K4) Analisar um determinado projeto e determinar as atividades de gerenciamento de testes que enfatizam o planejamento de testes, monitoramento de testes, e controle de testes

1.3 Teste Baseados em Risco

TM-1.3.1 (K2) Explicar as várias medidas que os testes baseados em riscos tomam para responder aos riscos

TM-1.3.2 (K2) Dar exemplos de diferentes técnicas que um Gerente de Teste pode usar para identificar riscos relacionados à qualidade do produto

TM-1.3.3 (K2) Resumir os fatores que determinam os níveis de risco relacionados à qualidade do produto

TM-1.3.4 (K4) Selecionar atividades de teste apropriadas para mitigar riscos de acordo com seu nível de risco em um determinado contexto

TM-1.3.5 (K2) Diferenciar entre exemplos pesados e leves de técnicas de teste baseadas em risco baseados em riscos

TM-1.3.6 (K2) Dar exemplos de métricas de sucesso e dificuldades associadas a testes baseados em riscos

1.4 A Estratégia de Teste do Projeto

TM-1.4.1 (K2) Explicar as escolhas típicas de uma abordagem de teste

TM-1.4.2 (K4) Analisar uma estratégia de teste organizacional e o contexto do projeto para selecionar a abordagem de teste apropriada

TM-1.4.3 (K3) Usar a metodologia de metas S.M.A.R.T. para definir objetivos de teste mensuráveis e critérios de saída

1.5 Aprimoramento do Processo de Teste

TM-1.5.1 (K2) Explicar como usar o modelo IDEAL para aprimorar o processo de teste em um determinado projeto

TM-1.5.2 (K2) Resumir a abordagem de melhoria baseada em modelos para testar a melhoria de processos e entender como aplicá-la em um contexto de projeto

TM-1.5.3 (K2) Resumir a abordagem de melhoria baseada em análise para testar a melhoria do processo e entender como aplicá-la em um contexto de projeto

TM-1.5.4 (K3) Implementar uma retrospectiva de projeto ou iteração para avaliar os processos de teste e descobrir áreas de teste a serem aprimoradas

1.6 Ferramentas de Teste

TM-1.6.1 (K2) Resumir as práticas recomendadas para a introdução de ferramentas

TM-1.6.2 (K2) Explicar o impacto de diferentes aspectos técnicos e comerciais ao decidir sobre um tipo de ferramenta

TM-1.6.3 (K4) Analisar uma determinada situação para criar um plano de seleção de ferramentas, abrangendo riscos, custos e benefícios

TM-1.6.4 (K2) Diferenciar os estágios do ciclo de vida da ferramenta

TM-1.6.5 (K2) Dar exemplos de coleta e avaliação de métricas e avaliação usando ferramentas

1.1 O Processo de Teste

Introdução

O *ISTQB® Foundation Level Syllabus v4* descreve um processo de teste que inclui as seguintes atividades: planejamento de teste, monitoramento e controle de testes, análise de teste, modelagem de teste, implementação de teste, execução de teste e conclusão de teste.

O *ISTQB® Foundation Level Syllabus v4* afirma que essas atividades no processo de teste são frequentemente implementadas de forma iterativa ou paralela, dependendo do modelo de ciclo de vida de desenvolvimento de software (SDLC) e do contexto do projeto. Geralmente, é necessário adaptar essas atividades ao contexto do produto e do projeto.

Neste syllabus, o foco está nas seguintes atividades principais de gerenciamento de testes:

- **Planejamento do teste:** definição dos objetivos do teste, abordagem do teste, escopo do teste, recursos do teste, cronograma do teste, resultados do teste e participantes do teste (stakeholders do teste).
- **Monitoramento e controle de testes:** monitoramento do progresso do teste, dos resultados do teste e dos desvios do teste; tomada de ações corretivas quando necessário; comunicação do status e dos resultados do teste aos stakeholders relevantes.
- **Conclusão do teste:** finalizar e arquivar os artefatos de teste, avaliar o processo de teste e o produto do teste, identificar as ações de melhoria do processo de teste e comunicar o encerramento do teste aos stakeholders relevantes.

A norma ISO/IEC/IEEE 29119-2 define os processos de gerenciamento de testes que abrangem essas atividades de gestão dos testes. Esses processos de gerenciamento de testes podem ser aplicados em diferentes níveis de teste,

como projeto, programa ou portfólio. Cada nível de teste pode ter seu próprio plano de teste que se alinha ao plano de teste de nível superior.

1.1.1 Atividades de Planejamento de Teste

Esta seção concentra-se nas atividades de planejamento de testes para diferentes escopos, como o projeto inteiro, um nível de teste, um tipo de teste ou uma versão/iteração/sprint no Ágil. Dependendo do escopo, o planejamento de testes pode começar e terminar em diferentes pontos do processo de desenvolvimento. O planejamento de testes é uma atividade que envolve a identificação das atividades e dos recursos necessários para atingir os objetivos de teste identificados na política de testes. O planejamento de testes deve ser iniciado o mais cedo possível no processo de desenvolvimento, de preferência antes da identificação dos requisitos, e deve ser atualizado à medida que o projeto avança. O planejamento de testes geralmente é um processo iterativo que requer replanejamento durante o projeto para acomodar mudanças e feedback.

As tarefas a seguir fazem parte do planejamento de testes (semelhantes às encontradas na ISO/IEC/IEEE 29119-2):

- **Entender o contexto e organizar o planejamento do teste:** Compreender o contexto da organização (p. ex., a política de teste e a estratégia de teste organizacional), o escopo do teste e o item de teste (ou seja, o produto de trabalho que está sendo testado) é fundamental para o planejamento do teste (consulte a seção 1.2., *O Contexto do Teste*). Isso também envolve todas as atividades necessárias para organizar o plano de teste e para obter a aprovação dos stakeholders (p. ex., Product Owner, Gerente de Projeto ou o Gerente de Desenvolvimento) dessas atividades e do cronograma.
- **Identificar e analisar os riscos do produto:** A análise de riscos envolve a identificação e a avaliação do impacto potencial e da probabilidade de riscos do produto como parte do planejamento de testes. Consulte a seção 1.3 deste syllabus, *Teste Baseado em Risco*, para obter mais detalhes sobre os riscos do produto.
- **Identificar abordagens de tratamento de riscos:** Com base na análise de riscos, as abordagens adequadas de tratamento de riscos são selecionadas e documentadas no plano de teste. Essas abordagens podem incluir ações preventivas, corretivas ou atenuantes para lidar com os riscos identificados.
- **Definir a abordagem de teste, estimar e alocar recursos de teste:** Com base na estratégia de teste organizacional nos padrões regulatórios, nas restrições impostas pelo projeto e nas abordagens de tratamento de riscos, a abordagem de teste é definida para o escopo atual do teste (consulte a seção 1.4, *A Estratégia de Teste do Projeto*). Quando uma abordagem de teste é definida, é importante estimar os recursos necessários de teste, como a equipe de teste, as ferramentas de teste, os ambientes de teste e os dados de teste, e alocar esses recursos para as atividades de teste.
- **Estabelecer o plano de teste:** O plano de teste deve ser aceito por todos os stakeholders e, portanto, as divergências entre eles devem ser resolvidas.

1.1.2 Atividades de Monitoramento e Controle de Teste

Para que o Gerenciamento de Testes proporcione um controle eficiente dos testes, é necessário estabelecer um cronograma e uma estrutura de monitoramento para permitir o acompanhamento do status e do progresso dos testes. Essa estrutura deve incluir as medidas e metas detalhadas necessárias para relacionar o status dos produtos e recursos de trabalho de teste ao plano e aos objetivos estratégicos.

Em projetos pequenos e menos complexos, pode ser relativamente fácil relacionar os produtos e as atividades de trabalho de teste ao plano e aos objetivos estratégicos, mas, em geral, é necessário definir objetivos mais detalhados para atingir esse objetivo. Isso pode incluir a definição de medidas e metas necessárias para atender aos objetivos de teste e à cobertura da base de teste.

É importante a necessidade de relacionar o status dos produtos e das atividades de trabalho de teste de uma maneira que seja compreensível e relevante para os stakeholders do projeto e do negócio.

Monitoramento e controle de testes são atividades contínuas.

O controle de teste compara o progresso real com o plano de teste e implementa ações corretivas conforme necessário. Ele orienta o teste para atender às estratégias e aos objetivos do teste (consulte a seção 1.4, *A Estratégia de Teste do Projeto*) e revisa as atividades de planejamento do teste quando necessário. Os dados de controle exigem informações detalhadas de planejamento de teste para reações apropriadas. Essa atividade envolve:

- Implementar o plano de teste e as diretrizes de controle
- Gerenciar os desvios dos testes planejados
- Tratar os riscos recém-identificados e alterados
- Estabelecer a prontidão para iniciar o teste
- Conceder e obter aprovação para a conclusão do teste com base nos critérios de saída

O monitoramento de teste envolve a coleta e o registro dos resultados dos testes, a identificação de desvios dos testes planejados, a identificação e a análise de novos riscos que exigem testes e o monitoramento de mudanças nos riscos identificados.

1.1.3 Atividades de Conclusão do Teste

A conclusão do teste geralmente ocorre em marcos do projeto (p. ex., uma versão, o fim de uma iteração ou a conclusão do nível de teste). Para quaisquer defeitos não resolvidos, são criadas solicitações de alteração ou itens do backlog do produto. Consulte o *ISTQB® Foundation Level Syllabus v4*. Quando os critérios de saída forem atendidos, os principais resultados deverão ser capturados, arquivados e fornecidos aos stakeholders relevantes. A conclusão do teste requer as seguintes tarefas:

- **Criar e aprovar o relatório de conclusão do teste:** essa tarefa garante que todos os testes tenham sido realizados e que todos os objetivos de teste tenham sido alcançados. Essa tarefa envolve a coleta de informações relevantes de vários materiais de teste, como planos de teste, resultados de teste, relatórios de progresso de teste, relatórios de conclusão de teste e relatórios defeitos. As informações coletadas são avaliadas e resumidas no relatório de conclusão do teste. O relatório de conclusão do teste é aprovado e comunicado aos stakeholders relevantes.
- **Arquivar testware:** essa tarefa identifica o material de teste que pode ser útil no futuro ou que se espera que seja reutilizado, que normalmente são os casos de teste. Ela os torna acessíveis e fáceis de entender para futura reutilização. Além disso, os resultados dos testes, os registros de testes, os relatórios de testes e outros materiais de teste devem ser arquivados temporariamente no sistema de gerenciamento de configuração.
- **Transferência de material de teste:** essa tarefa entrega os produtos de trabalho valiosos para aqueles que precisam deles. Por exemplo, defeitos conhecidos adiados ou aceitos devem ser comunicados àqueles que usarão ou apoiarão o uso do material de teste.
- **Executar todas as tarefas necessárias para limpar o ambiente de teste e restaurá-lo a um estado predefinido:** essa tarefa garante que o ambiente de teste esteja pronto para o próximo ciclo de testes ou projeto. Ela envolve a remoção de quaisquer dados de teste, ferramentas de teste, drivers de teste, simuladores de teste e scripts de teste do ambiente de teste. Também envolve a restauração do ambiente de teste para seu estado original ou desejado.
- **Executar/coletar/documentar lições aprendidas:** essa tarefa é realizada em retrospectivas onde as lições importantes aprendidas durante o processo de teste são discutidas e documentadas. Isso pode incluir descobertas em todo o ciclo de vida de desenvolvimento de software (SDLC). As lições aprendidas podem ser usadas para aprimorar o processo de teste, conforme descrito na seção 1.5, *Melhoria do Processo de Teste*.

1.2 O Contexto do Teste

Introdução

O contexto do teste abrange as condições e restrições exclusivas que influenciam o processo de teste, moldando as decisões e estratégias de planejamento, modelagem e execução de testes. É fundamental que os gerentes de teste compreendam esse contexto para alinhar os testes com as necessidades e os objetivos específicos do projeto de desenvolvimento de software. Esse contexto pode ser diferente com base no tipo de produto, no setor, nos requisitos regulatórios e, principalmente, no ciclo de vida de desenvolvimento de software (SDLC) que está sendo empregado.

Os gerentes de teste têm a tarefa de aplicar estratégias de teste estabelecidas e escolher técnicas de teste em vez de desenvolvê-las. Eles desempenham um papel fundamental na formulação de planos de teste adaptados ao contexto do projeto. Ao compreender e considerar esses vários fatores, os gerentes de teste podem garantir que o teste seja pertinente, eficaz e eficiente para atender aos objetivos do teste.

1.2.1 Stakeholders no teste

Os stakeholders no teste são indivíduos ou grupos com interesse direto ou indireto na qualidade do produto. Abaixo está uma lista típica de possíveis stakeholders, alterada para refletir seus diversos interesses em testes:

- **Desenvolvedores, líderes de desenvolvimento e gerentes de desenvolvimento:** além de implementar o sistema em teste e agir com base nos resultados do teste, esses stakeholders também estão envolvidos no teste de unidade e contribuem para o processo de teste.
- **Testadores, líderes de teste e gerentes de teste:** esses indivíduos preparam o material de teste, o que inclui o desenvolvimento de planos de teste e a contribuição para o processo de teste por meio de atividades como análise de requisitos, modelagem de teste, execução de teste, rastreamento e relatório de defeitos, automação de teste e relatório de progresso de teste.
- **Gerentes de projeto, Product Owners e usuários corporativos:** eles especificam os requisitos, definem o nível de qualidade solicitado e recomendam a cobertura necessária com base nos riscos percebidos. Eles também analisam os produtos de trabalho, participam do Teste de Aceite do Usuário (UAT) e tomam decisões com base nos resultados dos testes.
- **Equipe de operações:** envolvidos em testes de aceite operacional, eles garantem a prontidão do sistema para a produção e contribuem para a definição de requisitos não funcionais.
- **Clientes e usuários:** os clientes compram o produto, enquanto os usuários o utilizam diretamente. Ambos são fundamentais na definição dos requisitos e devem estar envolvidos no Teste de Aceite do Usuário (UAT) para validar se o produto atende às suas necessidades.

Essa lista não inclui todos os possíveis stakeholders. Os gerentes de teste devem realizar uma análise dos stakeholders como parte da criação da estratégia de teste e do plano de teste, considerando o escopo do teste na identificação de stakeholders específicos para o projeto.

1.2.2 Importância do conhecimento dos stakeholders no gerenciamento de testes

No gerenciamento de testes, é fundamental considerar as perspectivas e a influência de vários stakeholders. A matriz dos stakeholders, geralmente chamada de *matriz de poder e interesse*, orienta os gerentes de teste na priorização do envolvimento dos stakeholders e no gerenciamento eficiente das expectativas. A matriz de stakeholders é uma ferramenta estratégica no gerenciamento de testes que:

- Utiliza a experiência dos stakeholders, com usuários finais e equipes técnicas fornecendo feedback e percepção sobre performance e segurança.
- Apoia o gerenciamento de riscos ao destacar os interesses e a influência dos stakeholders, incentivando esforços proativos de mitigação.

- Valoriza as diversas perspectivas com feedback valioso.

A matriz dos participantes é composta de quatro quadrantes:

- **Promotores (alta influência, alto interesse):** colaboradores-chave com alta influência e interesse, vitais para moldar a estratégia e o plano de teste. e do plano.
- **Latentes (alta influência, baixo interesse):** Embora possam não ter um grande interesse nas tarefas cotidianas, suas decisões são essenciais para a alocação de recursos e a direção de alto nível do projeto.
- **Defensores (baixa influência, alto interesse):** Eles geralmente fornecem feedback qualitativo e podem ser mantidos engajados por meio de atualizações regulares e envolvimento em discussões específicas.
- **Apáticos (baixa influência, baixo interesse):** Embora não estejam intimamente envolvidos, atualizá-los sobre marcos significativos e buscar sua opinião sobre questões específicas pode gerar percepções únicas.

A função do Gerente de Teste inclui a compilação de uma lista detalhada de stakeholders e a compreensão da conexão de cada uma delas com as atividades de teste, usando a matriz de stakeholders para aumentar a eficácia das práticas de gerenciamento de testes.

Figura 1. Diferentes tipos de stakeholders

1.2.3 Gerenciamento de testes em um modelo de desenvolvimento de software híbrido

Os modelos híbridos de desenvolvimento de software integram elementos das abordagens sequenciais tradicionais e das práticas ágeis para atender às necessidades específicas do projeto ou às transições organizacionais. Os motivos a seguir são comuns para usar um modelo híbrido de desenvolvimento de software, embora, dependendo da organização e do projeto, também possam existir outros motivos:

- **Híbrido como transição para o Ágil:** a transição de metodologias tradicionais para metodologias Ágeis pode ser desafiadora devido às mudanças fundamentais no fluxo de trabalho, na cultura e na dinâmica da equipe. Os modelos híbridos oferecem uma abordagem equilibrada que facilita essa transição, combinando a estrutura dos métodos tradicionais com a flexibilidade das práticas ágeis.

- **Híbrido como adequado à finalidade:** algumas organizações ou projetos podem não ser capazes de migrar para o Ágil. Os projetos de alto risco podem exigir tarefas sequenciais em algum momento, e práticas ágeis em outros. Eles podem usar um modelo híbrido que se adapte ao seu objetivo.

Em uma configuração híbrida, as atividades de gerenciamento de testes podem incluir:

- Avaliar a compreensão e a capacidade da equipe de fazer uma transição perfeita entre as metodologias tradicionais e ágeis.
- Identificar os pontos fortes e fracos na adaptação a uma abordagem híbrida.
- Garantir que a equipe seja capaz de combinar processos estruturados com a flexibilidade do Ágil.
- Aprimorar a colaboração entre a equipe de testes e os stakeholders para gerenciar melhor os testes em sprints e fases de testes tradicionais.
- Participar de esforços coordenados, como *scrum-of-scrums* para testadores, para manter o foco nos testes e, ao mesmo tempo, contribuir para os objetivos gerais de desenvolvimento.
- Acompanhar e revisar os esforços de teste e os casos dentro dos sprints para garantir que estejam alinhados com as práticas ágeis.

Mais informações podem ser encontradas em (Fowler, 2010)

1.2.4 Atividades de gerenciamento de testes para vários modelos de ciclo de vida de desenvolvimento de software

Para alinhar adequadamente os testes dentro do modelo de SDLC, o Gerente de Teste deve compreender os vários modelos de SDLC usados em sua organização e utilizar esse conhecimento para alinhar adequadamente os testes com as atividades de desenvolvimento.

A tabela abaixo mostra uma comparação entre várias atividades de gerenciamento de testes com base em diferentes modelos de SDLC:

Aspecto	Modelo de desenvolvimento sequencial Por exemplo, Modelo V	Modelo de desenvolvimento iterativo por exemplo, SCRUM
Estimativa	Estimativa detalhada antecipada para cada nível de teste.	Estimativa iterativa, parte do planejamento da história por iteração.
Software de teste	Inclui estratégia, plano, casos, cronograma e relatórios.	Concentrar-se nos critérios de aceite e na definição do que foi feito; documentação mínima.
Funções	O Gerente de Teste supervisiona as decisões e o gerenciamento da equipe.	As funções são integradas; o facilitador ou treinador substitui o Gerente de Teste tradicional.
Ferramentas	Predominantemente, ferramentas de gerenciamento de testes adequadas para testes baseados em fases.	As ferramentas para CI/CD e automação são centralizadas, dando suporte a testes contínuos.
Abordagem de teste	Programado com antecedência, correspondendo às fases do projeto	Incorporado às iterações, com foco na adaptabilidade e no feedback.
Automação de testes	Implementado estrategicamente, pode ocorrer em vários estágios	Integrado desde o início, com ênfase na regressão automatizada em CI/CD
Monitoramento e relatórios	Relatórios baseados em marcos, com painéis automatizados opcionais	Relatórios contínuos com painéis de controle em tempo real e atualizações diárias de status

Aspecto	Modelo de desenvolvimento sequencial Por exemplo, Modelo V	Modelo de desenvolvimento iterativo por exemplo, SCRUM
Métricas	Concentra-se nas métricas de teste tradicionais e no gerenciamento de defeitos. (p. ex., execução de testes, taxas de defeitos).	Inclui métricas Ágil para rastreamento de iteração, além das métricas tradicionais. (p. ex., velocidade da equipe, gráficos de burndown)

Tabela 1: Atividades de gerenciamento de testes para vários modelos de SDLC

1.2.5 Atividades de gerenciamento de testes em vários níveis de teste

Teste de Componentes:

- Definir o escopo, os objetivos e os critérios de conclusão dos testes de componentes (Testes de Unidade).
- Envolver os testadores em atividades que vão além das funções tradicionais de teste, como revisões de código, em que suas habilidades analíticas agregam valor.
- Coordenar com a equipe de desenvolvimento a resolução de problemas e a contribuição para os testes unitários.

Teste de Integração de Componentes:

- Determinar as sequências de integração e as combinações de testes em colaboração com a equipe de desenvolvimento, levando em conta o modelo, as ferramentas e os processos do SDLC.
- Supervisionar o progresso para garantir que ele se alinhe às estratégias de teste de sistema e aceite.
- Gerenciar essa fase de forma cooperativa com os desenvolvedores, considerando também o teste de integração de componentes (unidade).

Teste de Integração do Sistema:

- Garantir que o escopo e os objetivos dos testes de integração do sistema sejam claros e estejam em sintonia com a avaliação de riscos e as metas de qualidade.
- Manter a supervisão do progresso, dos resultados e do gerenciamento de problemas durante os testes de integração do sistema.

Teste de Sistema:

- Adaptar o planejamento ao modelo SDLC, com alocação cuidadosa de recursos, seleção de ferramentas e agendamento.
    - Para projetos ágeis, integrar o teste do sistema com o teste iterativo de história, evitando fases de teste distintas e garantindo que o teste seja contínuo e integrado. Nos modelos sequenciais, os testes podem seguir estágios planejados.

Teste de Aceite:

- Colaborar com os stakeholders para analisar e confirmar o cumprimento dos critérios de aceite e planejar as atividades de teste, incluindo o gerenciamento de testes de usuários em UAT.
- Coordenar a logística dos testes de aceite, facilitando os testes nas instalações do cliente, para garantir que o produto atenda às necessidades de negócio e aos padrões de qualidade fora do ambiente de desenvolvimento.
- Facilitar a resolução de quaisquer problemas com o UAT e orientar os stakeholders no processo de aprovação do produto após o cumprimento dos critérios de aceite.

1.2.6 Atividades de gerenciamento de testes para diferentes tipos de testes

O gerenciamento eficaz de testes requer uma abordagem integrada que considere as demandas exclusivas de testes funcionais e não funcionais, testes caixa preta e caixa branca. Para os gerentes que realizam testes funcionais, o foco é garantir que todas as funcionalidades sejam completamente testadas e atendam aos requisitos definidos. O gerenciamento de testes não funcionais gira em torno da verificação dos atributos do sistema, como performance e segurança. O gerenciamento de testes caixa preta envolve a garantia de que os testes sejam focados no usuário e que todas as interações externas possíveis sejam cobertas. O gerenciamento de testes caixa branca enfatiza a compreensão da estrutura do código e a garantia de que os testes cubram completamente a lógica interna.

Gerenciamento de Testes Funcionais:

- *Planejamento estratégico e acompanhamento do progresso*: elaboração de uma estratégia de teste detalhada que se alinhe aos requisitos funcionais e aos objetivos do projeto, além do monitoramento do progresso.
- *Coordenação de recursos*: alocação eficiente de recursos humanos e técnicos para cobrir todos os aspectos funcionais do sistema.

Gerenciamento de Testes Não Funcionais:

- *Benchmarking de performance*: estabelecer padrões de referência de performance e gerenciar as atividades de teste que avaliam o sistema em relação a esses critérios.
- *Verificação de conformidade*: supervisionar os testes que garantem que o sistema atenda aos padrões não funcionais, como segurança, usabilidade e confiabilidade.

Gerenciamento de Testes Caixa Preta:

- *Análise da cobertura de teste*: garantir que os testes caixa preta cubram todos os cenários de usuário e requisitos de negócio.
- *Incorporação de feedback*: gerenciar o processo de coleta de feedback dos stakeholders para refinar as abordagens de teste caixa preta e a correção de defeitos.

Gerenciamento de Testes Caixa Branca:

- *Otimização da cobertura de código*: supervisionar o uso de ferramentas de cobertura de código para identificar lacunas nos testes caixa branca e direcionar recursos para tratar dessas áreas.
- *Integração de insights técnicos*: gerenciar a incorporação de percepções técnicas no processo de planejamento de testes, garantindo que os testes sejam modelados com uma compreensão do funcionamento interno do aplicativo.

1.2.7 Atividades de gerenciamento de testes para planejar, monitorar e controlar

Um gerenciamento de teste eficaz é a pedra angular de qualquer esforço de teste bem-sucedido, abrangendo uma ampla gama de atividades que necessitam de planejamento cuidadoso, monitoramento constante e controle estratégico. Os gerentes de teste desempenham um papel fundamental para garantir que o processo de teste não seja apenas eficaz e eficiente, mas também adaptado às demandas exclusivas do projeto em questão.

Planejamento de Teste:

- **Definição abrangente do escopo**: um plano de teste deve ser meticulosamente elaborado, incorporando uma definição completa do escopo. Isso inclui a identificação de todos os requisitos funcionais e não funcionais para garantir a cobertura completa do teste. Também envolve considerar as implicações das metodologias de teste caixa preta e caixa branca, garantindo que os casos de teste desenvolvidos sejam capazes de validar o sistema em teste de todos os ângulos.
- **Avaliação de riscos e plano de mitigação**: parte integrante do plano de teste é uma estrutura robusta de gerenciamento robusto de riscos. Os gerentes de teste devem realizar uma análise de risco detalhada, identificando possíveis vulnerabilidades e desafios que possam afetar tanto o fluxo de trabalho do projeto

quanto o produto final. O desenvolvimento de estratégias de mitigação é crucial, envolvendo um planejamento preventivo para contornar ou minimizar esses riscos de forma eficaz.

- **Estratégia de alocação de recursos**: o planejamento de recursos é outro elemento fundamental. Isso vai além da mera alocação para definir a estrutura da equipe, delinear funções e estabelecer protocolos de comunicação. Em ambientes em que as equipes são distribuídas, como nos modelos *onsite/offsite*, isso se torna especialmente importante para manter a sincronia e garantir uma colaboração contínua.

Monitoramento de Teste:

- **Supervisão da execução**: o monitoramento desempenha uma função central no processo de gerenciamento de testes. Ele envolve uma revisão contínua da execução do teste em relação ao plano estabelecido, acompanhando o progresso dos casos de teste e gerenciando os defeitos que surgirem. O ajuste das prioridades de teste com base nas avaliações de risco e nos desenvolvimentos em tempo real garante que os testes permaneçam focados e alinhados com as áreas mais críticas.
- **Otimização de ferramentas e ambientes**: a seleção e o uso criteriosos de ferramentas e ambientes de teste são fundamentais para apoiar a estratégia de teste. O monitoramento contínuo garante que eles sejam efetivamente integrados ao pipeline de CI/CD, facilitando os testes contínuos e os ciclos de feedback imediatos, que são vitais para o processo de desenvolvimento ágil.
- **Colaboração no desenvolvimento**: manter um relacionamento de trabalho próximo com a equipe de desenvolvimento é essencial para obter resultados bem-sucedidos de teste. Essa colaboração deve apoiar uma abordagem abrangente de testes, aproveitando as percepções das perspectivas caixa branca e caixa preta para abordar preventivamente os possíveis problemas.

Controle de Teste:

- **Gerenciamento de processos adaptativos**: o controle de testes consiste em ajustar dinamicamente o processo de testes em resposta a novas percepções, desafios e dinâmicas de projeto em evolução. Isso exige que o Gerente de Teste seja ágil e flexível, capaz de implementar mudanças na abordagem de testes que reflitam o estado atual do projeto.
- **Gerenciamento da qualidade**: Uma abordagem estruturada para o gerenciamento da qualidade é fundamental. Isso inclui a definição do que constitui a qualidade dentro do ciclo de vida dos testes e a tomada de decisões informadas sobre o progresso da fase de testes, o que é fundamental para manter a integridade do produto.

Concentrando-se nessas atividades específicas do planejamento, monitoramento e controle de testes, os gerentes de testes podem garantir que o processo de testes seja bem definido, adaptável às mudanças do projeto, resultando em um produto que atenda aos requisitos do projeto e às expectativas dos stakeholders.

1.3 Teste Baseado em Risco

Introdução

Os testes baseados em riscos envolvem a identificação, a avaliação, o monitoramento e a mitigação dos riscos para conduzir os testes. Esses riscos são identificados por diversos stakeholders e podem ser usados para selecionar e priorizar testes. Quanto maior o nível de risco quanto mais alto o nível de risco, mais cedo o teste deve começar e mais intenso e prolongado deve ser o esforço de teste.

1.3.1 Testes como uma atividade de mitigação de riscos

Um risco de produto é uma situação potencial em que podem existir problemas de qualidade em um produto. Quando os testes revelam defeitos, eles ajudam a mitigar o risco do produto, pois proporcionam a conscientização dos defeitos e oportunidades de lidar com eles antes do lançamento. Quando os testes não encontram defeitos, os testes indicam que o nível de risco do produto é menor do que o esperado.

Entre outras coisas, o Gerente de Teste é responsável por fornecer uma avaliação correta e confiável da qualidade do produto. Isso requer um envolvimento ativo no gerenciamento de riscos do projeto com foco nos riscos do projeto relacionados à garantia de qualidade (p. ex., requisitos ambíguos que levam a grandes problemas na validação tardia, ambientes de teste insuficientes que obstruem a execução do teste).

Testes baseados em risco concentram os testes nos riscos à qualidade. Ele segue o processo genérico de gerenciamento de riscos que consiste nas seguintes atividades principais:

- *Análise de risco*, que consiste na identificação e avaliação de riscos
- *Controle de riscos*, que consiste no monitoramento e mitigação de riscos

Essas atividades principais são organizadas logicamente em ordem sequencial, mas podem se sobrepor.

Para concentrar os testes nos riscos à qualidade, estes devem ser identificados e avaliados. Para ser mais eficaz, a análise de riscos deve incluir diversos stakeholders. Por ser o principal stakeholder na análise de riscos à qualidade o Gerente de Teste deve entender e monitorar essas atividades, e ser capaz de moderá-las.

O monitoramento de testes deve incluir o monitoramento dos riscos. Além de monitorar a evolução dos riscos de qualidade conhecidos, ele deve incluir a análise de quaisquer novos riscos de qualidade e a atualização do registro de riscos.

O Gerente de Teste é uma das várias pessoas que conduzem a mitigação dos riscos à qualidade. A mitigação dos riscos é distribuída em várias atividades de teste. Por exemplo, os resultados da análise de risco de qualidade são usados no planejamento de testes para concentrar os testes nas áreas corretas usando as técnicas corretas. Na análise de teste, os níveis de risco orientam a seleção das condições de teste a serem cobertas. Na execução do teste, a priorização baseada em riscos rege a sequência da execução do teste.

1.3.2 Identificação de riscos de qualidade

A tarefa do Gerente de Teste é reunir os riscos dos stakeholders. Estes podem identificar os riscos à qualidade por meio de uma ou mais das técnicas a seguir:

- Entrevistas com especialistas;
- Avaliações independentes;
- Retrospectivas;
- Workshops de risco;
- Brainstorming;
- Listas de verificação;
- Referir-se a experiências passadas;

Ao envolver a mais ampla amostra possível de stakeholders, o processo de identificação de riscos geralmente identifica a maioria dos riscos significativos do produto. A identificação de quais stakeholders devem participar nesse estágio é muito importante. É essencial garantir que esta lista de participantes seja abrangente e acordada com o gerente de projeto. A identificação de riscos que não inclui os principais stakeholders pode ser muito problemática. O segredo é garantir que todos eles tenham a chance de participar. Se não puderem participar, devem pelo menos ter a chance de delegar a tarefa. Quando os principais stakeholders podem não estar representados, uma reunião inicial pode ser usada para determinar se eles estão ausentes.

Nos testes baseados em riscos é importante entender que o risco não é distribuído uniformemente nos objetos de teste. Por exemplo, os componentes voltados para o cliente de um aplicativo de autoatendimento podem ter riscos de usabilidade muito diferentes dos componentes de administração. Identificar os riscos individuais dos vários itens de teste é uma tarefa importante no planejamento do teste.

A identificação de riscos geralmente produz subprodutos (ou seja, a identificação de problemas que não são riscos do produto). Os exemplos incluem perguntas ou problemas gerais sobre o produto ou o projeto, ou problemas em

documentos referenciados, como requisitos e especificações de projeto. Os riscos do projeto também são frequentemente identificados como um subproduto do risco de qualidade da Qualidade, mas não são o foco dos testes baseados em riscos. O Gerente de Teste pode, muitas vezes, desempenhar um papel importante ao destacar esses subprodutos e deixar claro que a Qualidade é uma preocupação de todos. Requisitos ruins ou ausentes geralmente são uma indicação de um problema mais fundamental no planejamento e na preparação, já que a garantia de qualidade está envolvida em todo o SDLC.

1.3.3 Avaliação do risco de qualidade

Após identificar os riscos, eles podem ser avaliados. Avaliação dos riscos de qualidade inclui a sua categorização por tipo (risco do produto ou risco do projeto) e por características de qualidade afetadas.

A determinação do nível de risco normalmente envolve a avaliação, para cada item de risco, da probabilidade de ocorrência do risco e do impacto do risco. Os fatores que influenciam a probabilidade para riscos de qualidade incluem:

- Complexidade da tecnologia, das ferramentas ou da arquitetura do sistema;
- Maturidade da organização;
- Problemas de pessoal com habilidades, disponibilidade, motivação ou trabalho autônomo, incluindo o conhecimento do SDLC em uso;
- Conflito dentro da equipe;
- Problemas contratuais com fornecedores;
- Equipes distribuídas geograficamente;
- Fraca liderança gerencial ou técnica;
- Pressão de tempo, recursos, orçamento e gerenciamento;
- Falta de atividades iniciais de garantia de qualidade;
- Altas taxas de alteração da base de teste, do produto ou da equipe.

Os fatores que influenciam o impacto do risco incluem:

- Frequência de uso do recurso afetado;
- Criticidade do recurso afetado;
- Criticidade da meta de negócios afetada;
- Danos à reputação;
- Perda de renda comercial;
- Possíveis perdas financeiras, ecológicas ou sociais, ou responsabilidade;
- Sanções legais civis ou criminais;
- Problemas de interface e integração;
- Falta de soluções alternativas razoáveis;
- Necessidades de segurança.

O Gerente de Teste combina a probabilidade do risco e o seu impacto para determinar o nível de risco.

Se a análise de risco for baseada em dados de risco extensos e estatisticamente válidos, uma avaliação quantitativa é apropriada. Por exemplo, a probabilidade do risco pode ser expressa como uma porcentagem e o impacto do risco como um valor. Nesse caso, o nível de risco pode ser calculado como o produto desses dois fatores. Normalmente,

porém, a probabilidade e o impacto do risco só podem ser verificados qualitativamente em escalas ordinais, por exemplo, como muito alto, alto, médio, baixo ou muito baixo. Os valores da probabilidade e do impacto do risco são então combinados em uma matriz de risco para criar um nível de risco agregado. Esse nível de risco agregado deve ser interpretado como uma classificação qualitativa e relativa em uma escala ordinal.

A menos que a análise de risco se baseie em dados extensos e estatisticamente válidos, a análise será qualitativa, com base nas percepções subjetivas dos stakeholders sobre a probabilidade e impacto do risco.

1.3.4 Mitigação do risco de qualidade por meio de testes apropriados

No desenvolvimento de software, o teste é a atividade de mitigação de risco de qualidade mais importante e possibilita a redução da probabilidade de falhas. Outras possíveis medidas de mitigação de riscos incluem um plano de contingência (p. ex., fornecendo soluções alternativas), transferência de riscos para terceiros (p. ex., o fornecedor de um componente) ou aceite de riscos.

No **planejamento de teste**, o tempo e o esforço associados ao desenvolvimento e à execução de um teste devem ser proporcionais ao nível de risco: Os testes para níveis de risco mais altos devem começar mais cedo e usar técnicas de teste mais rigorosas, enquanto os testes para níveis de risco mais baixos podem começar mais tarde e devem usar técnicas de teste menos rigorosas. Para melhor mitigar o risco geral por meio de testes, o Gerente de Teste deve analisar os seguintes fatores contextuais e selecionar uma abordagem de teste apropriada:

- **Os itens de teste:** itens de teste diferentes em um objeto de teste podem ter níveis diferentes do mesmo tipo de risco, portanto, um objeto de teste não precisa ser testado com rigor uniforme.
- **As características de qualidade:** os riscos que afetam as características específicas da qualidade devem ser mitigados pelos tipos de teste associados que precisam de esforço de teste específico, ambientes de teste e habilidades de teste.
- **Os níveis de teste e os tipos de teste:** certos riscos só podem ser testados dinamicamente em níveis de teste específicos; outros por meio de testes estáticos (p. ex., análise estática e revisões de código para manutenção) ou por uma combinação de ambos (p. ex., por meio de uma revisão da arquitetura e testes dinâmicos do sistema integrado para detectar vulnerabilidades de segurança). Testar cada item de teste o mais cedo possível reduz o risco de encontrar defeitos críticos no final do ciclo de vida, o que causaria atrasos e custos mais altos de falhas internas.
- **O SDLC:** as atividades de teste têm seus próprios critérios específicos de entrada. Vários SDLCs os preenchem em momentos diferentes.
- **A equipe de teste:** as pessoas mais qualificadas devem testar os itens de teste com os níveis de risco mais altos.
- **Os requisitos regulatórios:** alguns padrões relacionados à segurança (p. ex., o padrão IEC 61508) prescrevem as técnicas de teste e a cobertura necessária com base no nível de integridade. O Gerente de Teste precisa garantir que esses padrões sejam seguidos.

Além disso, o nível de risco deve influenciar as decisões de controle de qualidade, como o uso de revisões de produtos de trabalho, como casos de teste, o nível de independência dos testes em relação ao desenvolvimento e a extensão dos testes de regressão realizados.

Durante o **monitoramento e controle de teste**, os testes baseados em riscos permitem a geração de relatórios sobre o progresso do teste em termos do nível de risco residual em qualquer momento. Isso ajuda a equipe de desenvolvimento e os stakeholders a monitorar e controlar o desenvolvimento do software, incluindo a tomada de decisões de lançamento, com base no nível de risco residual. Para isso, é necessário relatar os resultados dos testes em termos de riscos de uma forma que os stakeholders possam entender.

Durante a **implementação do teste**, a priorização do teste é baseada nos níveis de risco. Durante a execução do teste, isso garante a cobertura antecipada das áreas mais críticas e a mitigação dos riscos de nível mais alto.

- Em alguns casos, os testes são priorizados para execução em ordem estritamente decrescente dos níveis de risco que abrangem, começando pelo mais alto. Essa abordagem é chamada de *depth-first* e é apropriada quando é importante mitigar os riscos de nível mais alto o mais cedo possível.
- Como alternativa, pelo menos um teste para cada risco recebe a prioridade máxima. Todos os outros testes são priorizados com base em seus níveis de risco coberto. Essa abordagem é chamada de *breadth-first* e é adequada quando os stakeholders querem ter uma visão geral da qualidade do produto o mais cedo possível. Na prática, os testes geralmente começam com a abordagem *depth-first*, mas, à medida que o tempo se torna mais limitado, eles mudam para a abordagem *breadth-first*, testando todos os itens de risco restantes pelo menos uma vez.

Independentemente dos testes baseados em risco serem realizados em *breadth-first*, em *depth-first* ou combinados, o tempo alocado para os testes pode ser consumido sem que todos os testes planejados sejam executados. Nesse ponto, os testes baseados em riscos facilitam o fornecimento de uma recomendação justificada à gerência sobre a extensão dos testes ou o aceite do risco restante.

1.3.5 Técnicas para teste baseado em risco

Há várias técnicas específicas com diversos graus de formalidade para implementar testes baseados em riscos. A adequação de uma técnica depende de considerações sobre o projeto, o processo e o produto. Há dois tipos básicos de técnicas: pesadas ou leves. Em sistemas críticos para a segurança, as técnicas pesadas são usadas com muita frequência. Em aplicativos não críticos para a segurança, geralmente são empregadas técnicas leves.

As técnicas pesadas são formais, usando procedimentos definidos e documentação detalhada. Elas envolvem grupos amplos de stakeholders. A avaliação de risco dentro das técnicas pesadas usa fatores detalhados de probabilidade de risco e impacto do risco e fórmulas matemáticas para calcular a probabilidade e o impacto a partir desses fatores. Exemplos de técnicas pesadas são:

- **Análise de perigo:** estende o processo analítico para cima, tentando identificar os perigos subjacentes a cada risco.
- **Custo da exposição:** determina para cada item de risco de qualidade a probabilidade de uma falha, o custo de uma perda associada a uma falha típica, e o custo dos testes para tais falhas.
- **Failure Mode and Effect Analysis (FMEA) e suas variantes:** identifica os riscos à qualidade, suas possíveis causas e seus prováveis efeitos e, em seguida, atribui gravidade, prioridade e taxas de detecção.
- **Fault Tree Analysis:** relaciona possíveis falhas aos defeitos que podem causá-las e, em seguida, relacionar os erros que podem causar esses defeitos, continuando até que as principais causas sejam identificadas.

Em contrapartida, as técnicas leves são menos completas e exigem menos esforço da equipe de teste e dos stakeholders. Assim como as técnicas pesadas, elas também se baseiam no envolvimento dos stakeholders, usando os resultados da análise de risco como base para o planejamento do teste e as condições do teste. No entanto, o grupo de stakeholders pode não ser tão amplo, e os fatores de risco geralmente são reduzidos ao impacto e probabilidade de risco em uma escala ordinal. Algumas dessas técnicas, como o *Systematic Software Testing* (SST) (Craig & Jaskiel, 2002), só podem ser usadas quando as especificações de requisitos são fornecidas. Outras técnicas, incluindo *Pragmatic Risk Analysis and Management* (PRAM) (Black, 2009), *Product Risk Management* (PRISMA) (van Veenendaal, 2012), usam os requisitos e/ou outras especificações como entrada para a análise de risco, mas podem funcionar inteiramente com base na entrada dos stakeholders.

1.3.6 Métricas de sucesso e dificuldades associadas aos testes baseados em riscos

Em uma retrospectiva, a equipe de teste deve medir até que ponto percebeu os benefícios dos testes baseados em riscos. Em muitos casos, isso envolve responder a algumas ou a todas as perguntas a seguir por meio do uso de métricas e consultas:

- Os stakeholders relevantes foram envolvidas ou representadas na análise de risco?
- O envolvimento dos stakeholders na análise de risco foi adequado?

- Se houve incidentes críticos na produção que indicam que defeitos críticos escaparam, eles foram resolvidos?
- A maioria dos defeitos de alta prioridade foi encontrada no início da execução do teste?
- A equipe de teste foi capaz de explicar os resultados do teste aos stakeholders em termos de risco?
- Os testes ignorados tinham um nível mais baixo de risco associado do que os executados?

Na maioria dos casos, os testes bem-sucedidos baseados em riscos resulta em uma resposta afirmativa para todas essas perguntas. No longo prazo, devem ser definidas metas de melhoria de processo para métricas de sucesso, além de se esforçar para melhorar a eficiência do processo de análise de risco de qualidade processo de análise de risco de qualidade.

O gerenciamento de riscos geralmente encontra dificuldades inesperadas devido a complexidades que muitas vezes são ignoradas.

- **Dificuldade em avaliar o nível de risco:** a estimativa do impacto do risco e a probabilidade do risco pode ser muito difícil. Solução: use dados históricos e peça a avaliação dos principais stakeholders do projeto.
- **Um bom começo:** o estabelecimento e a manutenção de uma abordagem adequada de teste baseado em risco são frequentemente negligenciados em face da alta pressão de curto prazo para o sucesso. Solução: monitoramento regular e relatório de riscos para os stakeholders.
- **Déjà vu:** o mesmo conjunto de riscos está sendo levantado para cada projeto, o que leva à complacência em relação ao risco. Solução: envolver as pessoas certas na identificação dos riscos e mitigar apenas os riscos considerados importantes.
- **Os principais riscos estão sendo perdidos:** a causa principal desse problema geralmente se deve ao envolvimento de pessoas inexperientes ou inadequadas no processo. Solução: envolva as pessoas adequadas e treine-as.
- **Mudança de stakeholders:** os stakeholders podem mudar com o tempo e também podem surgir novos riscos, portanto, a análise de riscos é uma atividade contínua e iterativa e não deve ser realizada apenas uma vez no início.

1.4 A Estratégia de Teste do Projeto

Introdução

Ao longo deste syllabus, a estratégia de teste organizacional é considerada como dada. O desenvolvimento e a manutenção de uma estratégia de teste organizacional são considerados no contexto da norma ISO/IEC/IEEE 29119-3 (onde é chamada de "*Organizational Test Strategy*") e dos syllabus do *ISTQB® Expert Level - Test Management* e do *ISTQB® Certified Tester Ágil Test Leadership at Scale*.

Se não houver uma estratégia de teste organizacional ou se ela não abranger os aspectos necessários, o gerenciamento de testes deverá procurar esclarecer os detalhes que faltam com os stakeholders relevantes.

No contexto desta seção, a definição de uma estratégia de teste de projeto é um exemplo de qualquer tipo de estratégia de teste detalhada para um projeto, uma versão, um produto ou qualquer outro tipo de iniciativa de desenvolvimento ou aquisição de sistemas. Uma estratégia de teste de projeto (chamada de "*Test Strategy*" na ISO/IEC/IEEE 29119-3) descreve a abordagem de teste em um contexto específico para que os objetivos da organização possam ser atingidos, especialmente aqueles relacionados à qualidade do produto e às atividades de teste. Uma estratégia de teste também pode existir para um único nível de teste ou um tipo de teste.

A estratégia de teste do projeto é o principal resultado do planejamento de testes para um projeto e, normalmente, é documentada em um plano de teste ou como parte de outros documentos. Recomenda-se a documentação da estratégia de teste, mas não necessariamente na forma de um plano de teste formal. A necessidade de documentação depende do contexto do teste (consulte a seção 1.2, *Contexto do Teste*). Quando um projeto segue um modelo de desenvolvimento sequencial a estratégia de teste do projeto geralmente é documentada, de preferência no

plano de teste (consulte ISO/IEC/IEEE 29119-3). A documentação também é frequentemente exigida por contratos, acordos, órgãos reguladores ou leis.

1.4.1 Escolha de uma abordagem de teste

A estratégia de teste do projeto orienta todas as atividades de teste em um projeto e detalha objetivos, recursos, cronogramas e responsabilidades. Essa estratégia deve ser adaptada aos requisitos exclusivos do projeto. As principais decisões incluem a seleção de níveis de teste, tipos de teste e técnicas de teste para testes estáticos e dinâmicos e outras práticas de teste (p. ex., teste com script, teste manual, teste consecutivo).

Em teoria, todos os tipos de teste podem ser realizados em qualquer nível de teste, e qualquer técnica de teste pode ser aplicada a qualquer tipo de teste em qualquer nível de teste. Na prática, a seleção e a combinação adequadas dessas escolhas têm um impacto significativo na eficácia e na eficiência dos testes. Por exemplo, a capacidade de manutenção do código pode ser avaliada com mais eficácia e eficiência usando a análise estática do código ou a revisão do código. Por outro lado, a eficiência da performance pode ser mais bem avaliada por meio de testes de sistema com script devido à interação de componentes internos, ou a utilidade da funcionalidade pode ser mais bem validada com os usuários por meio de testes de aceite manuais desenvolvidos de forma colaborativa. A escolha da melhor abordagem para uma estratégia de teste pode ser um processo complexo que pode ser influenciado pela estratégia de teste organizacional, pelo contexto do projeto e por outros aspectos.

Portanto, a seleção e a combinação de níveis de teste, tipos de teste e técnicas de teste são essenciais para uma estratégia eficaz de teste de projeto, pois influenciam significativamente a eficiência e a eficácia do teste.

1.4.2 Análise da estratégia de teste organizacional, do contexto do projeto e de outros aspectos

A estratégia de teste organizacional, o contexto do projeto e os fatores ou restrições adicionais relacionados ao teste devem ser totalmente compreendidos para permitir o desenvolvimento de uma estratégia de teste do projeto.

Para escolher a abordagem de teste adequada, os seguintes fatores geralmente devem ser analisados:

- **Domínio:** o domínio para o qual o produto será criado ou modificado. Quaisquer normas, padrões e práticas específicas do domínio podem alterar o rigor dos testes, a documentação necessária e o nível de detalhes. Por exemplo, em produtos farmacêuticos e medicamentos, a abordagem de teste geralmente enfatiza o teste intensivo de aceite do usuário com foco nos riscos à saúde do paciente, usando casos de teste baseados em requisitos funcionais do usuário, enquanto o teste de aceite do usuário para aplicativos de seguro baseados na Web pode se concentrar na usabilidade e no aumento da probabilidade de novos contratos de seguro por meio de testes A/B.
- **Metas organizacionais e características de qualidade abrangentes:** As metas organizacionais podem incluir a necessidade de demonstrar o valor dos testes e aumentar o seu grau de automação ou as características de qualidade do processo de teste, como o nível de maturidade dos testes ou a eficiência da detecção de defeitos. Isso pode determinar os níveis de teste e os tipos de teste que precisam ser seguidos.
- **As metas do projeto e o tipo de projeto:** as metas do projeto (p. ex., com relação a orçamento, tempo e qualidade) e o tipo de projeto (ou seja, desenvolvimento de produto específico para o cliente ou orientado para o mercado) normalmente contêm restrições e riscos, bem como oportunidades que afetam os testes. Por exemplo, restrições orçamentárias e de tempo podem exigir o uso rigoroso de testes baseados em riscos para priorizar os casos de teste para a execução do teste, enquanto o desenvolvimento de um produto especificamente para um cliente pode exigir testes que englobem critérios de aceite contratual predefinidos.
- **Recursos de teste:** devem ser consideradas todas as restrições relacionadas à disponibilidade dos recursos de teste, incluindo as ferramentas de teste, a infraestrutura de teste, a tecnologia e o ambiente de desenvolvimento usados no projeto, bem como a equipe de teste disponível e suas habilidades. (consulte a seção 3.1, *A Equipe de Teste*). Por exemplo, o teste baseado na experiência requer testadores com bom conhecimento do domínio; os aplicativos móveis normalmente precisam ser testados em um número limitado de dispositivos diferentes; o uso de ferramentas de teste pode ser limitado pelo número de licenças disponíveis.

- **O modelo de ciclo de vida de desenvolvimento de software usado para o projeto:** para determinar os níveis de teste apropriados, o esforço de teste, os critérios de entrada e os critérios de saída apropriados, consulte o *ISTQB® Foundation Level Syllabus v4*, seções 2.2 e 5.1. Um ciclo de vida de software com integração contínua requer mais testes automatizados do que um desenvolvimento único usando um modelo em cascata e, portanto, diferentes tipos de teste e técnicas de teste podem ser usados.
- **Interfaces com outros sistemas:** em um sistema de sistemas, é essencial alinhar os testes com outras equipes ou projetos e selecionar níveis de teste adequados, especialmente para testes de integração de sistemas. Por exemplo, o teste baseado em risco ajuda a priorizar e dimensionar os testes de integração do sistema.
- **Disponibilidade de dados de teste:** devem ser consideradas as restrições quanto à disponibilidade dos dados de teste, como a necessidade de dados de teste anônimos da produção ou a criação de dados de teste específicos que podem ser difíceis de fornecer e precisam ser validados, como dados para testes de IA. Por exemplo, os testes baseados em modelos podem dar suporte à criação e ao gerenciamento de dados de teste.

O Gerente de Teste deve determinar qual combinação de técnicas de teste, níveis de teste e tipos de teste deve ser usada como a melhor abordagem para satisfazer a estratégia de teste da organização. O Gerente de Teste deve determinar qual combinação de técnicas de teste, níveis de teste e tipos de teste deve ser usada como a melhor abordagem para satisfazer a estratégia de teste organizacional, o contexto do projeto e fatores ou restrições adicionais relacionados ao teste.

1.4.3 Definição dos objetivos do teste

Um plano de teste deve ser definido para cada projeto de teste e deve conter, entre outros aspectos, o escopo do teste, os objetivos do teste e os critérios de saída. O plano de teste pode ser configurado no nível da versão, como um plano de teste de projeto (também conhecido como plano mestre de teste) e, se necessário, como um plano de teste de nível para os diferentes níveis de teste. Além disso, podem ser definidos planos de teste para as diferentes características de qualidade, como um plano de teste de segurança ou um plano de teste de performance. No desenvolvimento Ágil de software e no desenvolvimento híbrido de software um plano de teste de iteração pode ser acordado. Para cada versão e iteração, o escopo dos recursos funcionais e suas características não funcionais a serem entregues são definidos no plano de teste e acordados pelos stakeholders.

Associados aos recursos fornecidos para teste em um projeto, os objetivos de teste do projeto e os critérios de saída devem ser definidos. Isso pode ser feito usando a metodologia de metas S.M.A.R.T:

- **S**pecific = específico. O objetivo do teste do projeto e o critério de saída do projeto devem ser claros e não ambíguos.
- **M**easurable = mensurável. Deve ser quantificável e ter critérios específicos para medir o progresso e determinar se foi alcançado.
- **A**chievable = atingível. Deve ser viável considerando os recursos, o prazo e as capacidades disponíveis.
- **R**elevant = relevante. Deve estar alinhado com os objetivos gerais do projeto.
- **T**imely = oportuno. Deve ter um cronograma específico e um prazo definido para conclusão.

Os objetivos de teste do projeto devem abordar todos os aspectos de qualidade e quantidade visados, desde que sejam mensuráveis ou atingíveis. Exemplos de objetivos de teste do projeto são:

- Atingir os critérios de saída especificados dentro do período de tempo definido;
- Atender às metas de qualidade da organização (p. ex., medido como um indicador-chave de performance para o número de reclamações de clientes sobre um produto);
- Cumprir as regras e os regulamentos do setor específico;
- Garantir a disponibilidade de dados somente para usuários autorizados (p. ex., por direitos de acesso);

- Verificar a integridade funcional, a correção funcional, a performance, a eficiência, a portabilidade e a segurança da migração de dados;
- Aprimorar o nível de automação de testes (p. ex., para testes de regressão ou performance em uma porcentagem definida);
- Refatoração de código bem-sucedida e demonstração de que não introduziu novos defeitos (p. ex., para remover código-fonte mal estruturado ou dívida técnica e, ao mesmo tempo, manter a funcionalidade existente, comprovada por um teste de regressão);
- Comprovar a segurança das interfaces (p. ex., validando mensagens XML (*Extensible Markup Language*) em relação à definição do esquema XML para garantir a rejeição de dados maliciosos);
- Verificação da usabilidade de uma interface de usuário e obtenção de algum grau de subcaracterística (p. ex., medindo o tempo necessário para concluir uma tarefa específica em uma loja on-line).

Além da contagem e da medição dos objetivos de teste do projeto, deve-se considerar a avaliação do nível de qualidade por especialistas no nível de domínio e por stakeholders.

Dependendo do contexto do projeto e dos objetivos do teste, às vezes podem ser necessários vários ambientes de teste com os recursos e/ou as ferramentas de teste disponíveis. Os ambientes de teste podem não estar todos disponíveis ao mesmo tempo. Isso precisa ser considerado ao formular objetivos de teste e critérios de saída viáveis.

Dependendo do contexto do projeto, outros fatores deverão ser considerados na definição dos objetivos e do escopo do teste do projeto, conforme descrito na seção 1.2 deste syllabus, *Contexto do Teste*.

1.5 Aprimoramento do Processo de Teste

Introdução

Os testes são uma parte importante do desenvolvimento de software e geralmente representam pelo menos em torno de 30% a 40% do custo total do projeto. Além dos muitos desafios (técnicos) que os projetos de software enfrentam (p. ex., complexidade e tamanho crescentes, novas tecnologias, ampla variedade de dispositivos e sistemas operacionais e vulnerabilidades de segurança), é necessário otimizar a eficácia e a eficiência dos testes, bem como aprimorar os processos de teste. Aprender com as práticas recomendadas existentes e com os próprios erros permite aprimorar o processo de teste e tornar os projetos mais bem-sucedidos.

Um processo de melhoria em nível organizacional é normalmente mais útil do que um processo de melhoria no nível do projeto ou equipe. No entanto, também é possível e benéfico aplicar a melhoria do processo no nível do projeto ou da equipe, mas ela deve ser adaptada às necessidades do projeto ou da equipe. O aprimoramento de um teste pode ser iniciado, por exemplo, pela insatisfação com os resultados dos testes atuais, defeitos inesperados, mudanças nas circunstâncias, um resultado de referência ou falta de comunicação. Há diferentes técnicas disponíveis para aprimorar os testes (Bath & van Veenendaal, 2014). Algumas dessas técnicas são descritas a seguir. As técnicas descritas neste syllabus podem ser aplicadas tanto aos modelos de desenvolvimento sequencial quanto aos modelos de desenvolvimento de software ágil/incremental. O *ISTQB® Expert Level Improving the Test Process Syllabus* oferece uma visão mais profunda.

1.5.1 O processo de aprimoramento de testes (IDEAL)

Uma vez que tenha sido acordado que os processos de teste devem ser aprimorados, as atividades de implementação de aprimoramento de processos a serem adotadas para essa atividade podem ser definidas como no modelo IDEAL que se baseia em ideias semelhantes às do conhecido ciclo planejar-executar-verificar-agir (PDCA). IDEAL é um acrônimo que significa *Initiating* (Iniciar), *Diagnosing* (Diagnosticar), *Establishing* (Estabelecer), *Acting* (Agir) e *Learning* (Aprender).

Embora o IDEAL tenha sido originalmente definido para apoiar atividades de melhoria no nível organizacional, ele também pode ser aplicado no nível de projeto ou de equipe de desenvolvimento de software Ágil. Em um contexto de projeto, os objetivos das atividades (veja a seguir) ainda precisam ser alcançados. A principal diferença é

provavelmente a fase inicial, que é muito menor em um projeto ou equipe do que em um nível organizacional. O diagnóstico por meio de uma retrospectiva e o estabelecimento de um plano provavelmente seriam muito menores do que no nível organizacional. A atuação e o aprendizado também serão relevantes no nível do projeto ou da equipe.

Iniciando o processo de aprimoramento

No início do processo de melhoria, os objetivos e o escopo das melhorias do processo são acordados pelos stakeholders.

Diagnóstico da situação atual

O processo de teste atual é avaliado para identificar possíveis melhorias. Normalmente, a avaliação é feita em relação a uma estrutura padrão no caso de melhoria do processo de teste baseado em modelo (consulte a seção 1.5.2, *Melhoria do Processo de Teste Baseado em Modelos*) ou pode ser baseada em uma análise de métricas específicas no caso de melhoria do processo de teste com base analítica (consulte a seção 1.5.3, *Abordagem de Melhoria do Processo de Teste com Base Analítica*).

Estabelecimento de um plano de melhoria do processo de teste

Um plano de melhoria do processo de teste pode ser um documento formal que lista todas as ações detalhadas que devem ser executadas para obter melhorias. Dependendo do contexto, o plano pode ser altamente informal e muito leve. A lista de possíveis melhorias no processo deve ser priorizada. A priorização pode ser baseada no retorno sobre o investimento (ROI), nos riscos, no alinhamento com as estratégias do projeto ou da equipe e/ou nos benefícios quantitativos ou qualitativos mensuráveis que devem ser alcançados.

Agir para implementar a melhoria do processo de teste

O plano de melhoria do processo de teste para a entrega das melhorias é implementado. Normalmente, isso inclui treinamento e teste dos processos alterados e sua implementação completa no projeto ou na equipe.

Aprendendo com o programa de melhoria

Após a implementação completa das melhorias no processo, é essencial verificar quais benefícios, planejados ou inesperados, foram encontrados. Depois de saber o que funcionou e o que não funcionou, devemos agir com base nessas informações e, somente depois disso, o próximo ciclo de melhoria poderá ser iniciado.

1.5.2 Melhoria do processo de Teste Baseado em Modelos

Uma premissa tanto para a melhoria do processo de teste baseado em modelos e a melhoria baseada em análise é a suposição de que a qualidade do produto é altamente influenciada pela qualidade dos processos que estão sendo usados e aplicados. Ao aplicar a melhoria do processo de teste baseado em modelos, usa-se um modelo de melhoria de teste. Estes modelos baseiam-se nas melhores práticas de teste e organizam o a melhoria do teste de forma gradual.

Surgiram vários modelos de processos recomendados que apoiam o aprimoramento do processo de teste. Entre eles estão o *Test Maturity Model Integrated (TMMi)*® e *TPI NEXT*®.

A melhoria baseada em modelos também pode ser aplicada no nível do projeto. Nesses casos, a avaliação e o processo de melhoria concentram-se especificamente nos processos de teste ou nas principais áreas definidas no modelo que se relacionam com as atividades no nível do projeto (p. ex., planejamento de teste e modelagem de teste) e, muitas vezes, omitem em grande parte aqueles que estão no nível da organização (p. ex., política de teste e organização de teste). Como alternativa, também é possível adaptar adequadamente as práticas que abordam o nível organizacional ao contexto do projeto.

Para obter mais informações sobre a melhoria do processo de teste baseado em modelos consulte ISTQB® Expert Level Improving the Test Process Syllabus.

Test Maturity Model Integrated

O TMMi ® (van Veenendaal & Cannegieter, 2011) (van Veenendaal, 2020) é composto por cinco níveis de maturidade. Cada nível de maturidade, exceto o nível 1 do TMMi® , contém áreas de processo de teste e metas de melhoria. Além disso, para facilitar e apoiar sua implementação, o TMMi® contém práticas, subpráticas e exemplos. O TMMi® foi inicialmente desenvolvido para complementar o *Capability Maturity Model Integration (CMMI®)*, mas hoje é amplamente utilizado independentemente do CMMI®.

Para facilitar e apoiar a atualização do TMMi® no desenvolvimento ágil de software, foi desenvolvida uma diretriz específica que explica como o TMMi® pode ser usado e aplicado de forma benéfica no desenvolvimento ágil de software.

Para obter mais informações sobre o TMMi® , consulte www.tmmi.org.

TPI NEXT®

O modelo TPI NEXT® (van Ewijk, 2013) define 16 áreas-chave, cada uma das quais abrange um aspecto específico do processo de teste (p. ex., estratégia de teste, métricas de teste, ferramentas de teste e ambiente de teste). Quatro níveis de maturidade são definidos no modelo para cada uma das 16 áreas-chave.

Pontos de controle específicos são definidos para avaliar cada área-chave em cada um dos níveis de maturidade. Os resultados da avaliação são resumidos e visualizados por meio de uma matriz de maturidade que abrange todas as áreas-chave.

Para obter mais informações sobre o TPI NEXT® consulte www.tmap.net.

1.5.3 Abordagem de melhoria do processo de Teste com Base Analítica

Usando uma abordagem de melhoria baseada em modelos, conforme descrito na seção anterior, as melhorias são introduzidas comparando a abordagem de teste de um projeto ou equipe com as práticas recomendadas externas. As abordagens analíticas identificam problemas com base em dados do próprio projeto ou equipe. As melhorias apropriadas podem ser derivadas de uma análise desses problemas. As abordagens analíticas podem ser usadas em conjunto com uma abordagem baseada em modelos para verificar os resultados e proporcionar diversidade.

Os problemas podem ser identificados por meio de dados quantitativos e qualitativos. A seção 1.5.3 deste syllabus, *Abordagem de Melhoria do Processo de Teste com Base Analítica*, apresenta abordagens analíticas que usam principalmente dados quantitativos do processo de teste e dados de defeitos para avaliar a abordagem atual. A seção 1.5.4 deste syllabus, *Retrospectivas*, apresenta as retrospectivas, nas quais os dados qualitativos sobre o que funciona bem e o que não funciona bem são coletados dos membros da equipe de desenvolvimento e de teste.

A análise de dados é importante para a melhoria objetiva do processo de teste e um suporte valioso para avaliações puramente qualitativas, que, de outra forma, podem resultar em recomendações imprecisas que não são apoiadas por dados. A aplicação de uma abordagem analítica à melhoria geralmente envolve uma análise quantitativa do processo de teste para identificar áreas problemáticas e definir metas específicas do projeto. A definição e a medição dos principais parâmetros são necessárias para avaliar o processo de teste e verificar se as melhorias foram bem-sucedidas.

Exemplos de abordagens analíticas são:

- Análise de causa raiz
- Análise usando medidas, métricas e indicadores
- A abordagem GQM (Goal-Question-Metric)

A análise de causa raiz é o estudo de problemas para identificar suas causas raiz. Isso permite a identificação de soluções que eliminam as causas dos problemas, em vez de apenas abordar os sintomas óbvios imediatos. Um possível procedimento de análise envolveria a seleção de um conjunto apropriado de defeitos, a identificação de agrupamentos nesses dados e o uso de diagramas de causa e efeito (também chamados de diagramas de Ishikawa ou espinha de peixe) para identificar as causas básicas de agrupamentos de defeitos importantes. As melhorias são então derivadas para evitar a ocorrência de defeitos semelhantes.

Medidas, métricas e indicadores são usados de forma quantitativa para avaliar a performance do processo de teste no projeto ou na equipe. Os principais atributos do processo de teste a serem considerados são eficácia, eficiência e previsibilidade. Para cada um desses atributos, uma ou várias métricas podem ser selecionadas. Ao coletar e analisar os dados correspondentes, é possível identificar as principais áreas que exigem melhorias.

A abordagem GQM (Basili, et al., 2014) (van Solingen & Berghout, 1999) fornece uma estrutura para definir e analisar métricas que são adaptadas às necessidades de informações dos stakeholders do projeto. As metas de medição definem um aspecto de qualidade de um objeto que precisa ser medido para uma finalidade, perspectiva e contexto específicos. Essas metas são refinadas em perguntas que definem o aspecto da qualidade do ponto de vista dos stakeholders. Em seguida, são selecionadas as métricas que fornecem as informações necessárias para responder à pergunta. Os dados coletados para as métricas respondem às perguntas, para avaliar a meta de medição e satisfazer as necessidades de informação dos stakeholders.

Mais informações sobre essas abordagens de melhoria do processo de teste com base analítica podem ser encontradas no *ISTQB® Expert Level Improving the Test Process Syllabus*.

1.5.4 Retrospectivas

As retrospectivas são reuniões em que uma equipe analisa seus métodos e sua colaboração, capta as lições aprendidas (boas e ruins) e decide sobre mudanças e ações para obter melhorias (tanto para questões de teste quanto para questões não relacionadas a testes). As retrospectivas abordam tópicos como o processo, as pessoas, a organização, a colaboração e as ferramentas.

As retrospectivas são usadas tanto nos modelos de desenvolvimento sequencial quanto no desenvolvimento ágil de software. Nos modelos de desenvolvimento sequencial, elas fazem parte da conclusão dos testes. Nesse contexto, as retrospectivas têm o objetivo de gerar lições aprendidas para gerenciar melhor os projetos futuros. No desenvolvimento ágil de software, as retrospectivas geralmente são realizadas no final de cada iteração para discutir o que foi bem-sucedido e o que precisa ser melhorado, e como essas melhorias podem ser incorporadas na próxima iteração. As retrospectivas são realizadas por toda a equipe e, portanto, apoiam a abordagem de toda a equipe e promovem a melhoria contínua. Observe que, às vezes, são necessárias retrospectivas específicas para tratar de problemas de teste.

Uma retrospectiva típica consiste nas seguintes etapas:

Introdução: o objetivo e a agenda da retrospectiva são revisados, e uma atmosfera de confiança mútua é criada para que os problemas possam ser discutidos sem culpa ou julgamento.

Coleta de dados: os dados são coletados com relação ao que aconteceu durante a iteração ou o projeto. É possível coletar dados qualitativos, como uma linha do tempo dos principais eventos que identifica problemas, e lista como cada membro da equipe se sente em relação a esses problemas. Além disso, podem ser apresentados dados quantitativos de métricas, por exemplo, dados sobre o progresso do teste, detecção de defeitos, eficácia dos testes, eficiência dos testes e previsibilidade podem fornecer uma visão objetiva dos testes do projeto ou da iteração.

Derivar melhorias: os dados coletados são analisados para entender a situação atual e gerar ideias de melhoria. Por exemplo, a análise de causa-raiz pode ser aplicada para identificar as causas-raiz dos problemas identificados, e uma sessão de brainstorming pode ser realizada para gerar ideias sobre como resolver as causas-raiz.

Decidir sobre as ações de melhoria: as ações para implementar as ideias de melhoria são derivadas e priorizadas. Um plano de melhoria e responsabilidades são definidos. Metas e métricas associadas podem ser definidas para avaliar o impacto das ações sobre os problemas identificados. A implementação de muitas melhorias com etapas verificáveis ao mesmo tempo é difícil de gerenciar.

Retrospectiva de fechamento: nesta última etapa, a retrospectiva em si é revisada para identificar os pontos fortes e as melhorias no processo de retrospectiva. Uma retrospectiva é realizada regularmente, especialmente no desenvolvimento ágil de software. A melhoria contínua também é aplicada à própria retrospectiva.

É importante documentar adequadamente os resultados de uma retrospectiva. Em um modelo de desenvolvimento sequencial, os resultados, as conclusões, e as recomendações precisam ser distribuídos e comunicados de forma compreensível aos membros da organização. No desenvolvimento ágil de software, os problemas e as ações também devem ser documentados para permitir a revisão e seu possível impacto sobre os problemas na próxima iteração.

Os testadores, por fazerem parte da equipe (de projeto), trazem sua perspectiva única. Eles podem levantar problemas relacionados a testes (e outros) e estimular a equipe a pensar em possíveis melhorias.

Mais informações podem ser encontradas em (Derby & Larsen, 2006).

1.6 Ferramentas de Teste

Introdução

Há três tipos de ferramentas de negócio:

- Ferramentas comerciais
- Ferramentas de código aberto
- Ferramentas personalizadas

Ao selecionar uma ferramenta de negócio, todos os requisitos e normas organizacionais e dos stakeholders devem ser considerados.

Também existem ferramentas técnicas, como ferramentas de automação de testes, ferramentas de gerenciamento de testes e muitas outras.

Exemplos de uso de ferramentas de teste podem ser encontrados no *ISTQB® Foundation Level Syllabus v4*.

1.6.1 Boas práticas para a introdução de ferramentas

Esta seção contém as etapas necessárias para a avaliação e a introdução de uma ferramenta de teste.

Um Gerente de Teste pode estar envolvido na introdução de uma ferramenta ou pode promover ou facilitar o processo de introdução. Os gerentes de teste geralmente são responsáveis por uma ferramenta de teste dedicada ou por

qualquer ferramenta relacionada a testes, como uma ferramenta de gerenciamento de requisitos, gerenciamento de defeitos ou monitoramento.

Há boas práticas e considerações genéricas ao avaliar e selecionar uma ferramenta de teste. Essas práticas e considerações incluem o seguinte:

- Identificar oportunidades de aprimoramento de processos, com o apoio de ferramentas adequadas;
- Compreender as tecnologias usadas em uma organização e selecionar uma ferramenta compatível com essas tecnologias;
- Compreender como uma ferramenta é integrada técnica e organizacionalmente ao SDLC;
- Avalie a ferramenta em relação a requisitos claros e critérios objetivos;
- Avalie o fornecedor se estiver pensando em usar uma ferramenta comercial. Avalie o suporte para ferramentas não comerciais (p. ex., de código aberto);
- Identificar os requisitos internos para coaching, mentoring ou treinamento no uso da ferramenta
- Considerar os prós e contras de vários modelos de licenciamento;
- Como etapa final, realize uma avaliação de prova de conceito.

As boas práticas genéricas na adoção e implementação de uma ferramenta incluem:

- Execute um projeto piloto para validar os critérios e requisitos de seleção e para avaliar como a ferramenta se ajusta aos processos e práticas existentes;
- Adaptar e aprimorar os processos para que se ajustem ao uso da ferramenta e também adaptá-la aos processos existentes, se necessário;
- Definir diretrizes para o uso da ferramenta;
- Fornecer treinamento, orientação e mentoria para os usuários da ferramenta;
- Implementar a ferramenta na organização em incrementos;
- Implementar uma maneira de coletar informações do uso real da ferramenta para melhorias adicionais;
- Definir a propriedade da ferramenta.

1.6.2 Aspectos técnicos e comerciais das decisões sobre ferramentas

Vários fatores afetam a decisão sobre a implementação e o uso de uma ferramenta. Para um Gerente de Teste, é importante conhecê-los e abordá-los.

- **Normas e segurança**: As organizações que desenvolvem software de segurança crítica ou de missão crítica, ou que estão sujeitas à conformidade normativa, podem preferir ferramentas comerciais, pois elas atendem com mais frequência aos padrões exigidos e geralmente possuem a certificação adequada.
- **Aspectos financeiros:** As ferramentas de código aberto geralmente têm um custo inicial mais baixo devido ao suporte e ao desenvolvimento da comunidade. As ferramentas comerciais podem ter um preço de compra único, bem como custos de licença recorrentes. É difícil determinar o custo inicial de uma ferramenta personalizada porque ele depende dos requisitos e do estágio de desenvolvimento da ferramenta. Além dos custos iniciais, o custo de treinamento e manutenção durante a vida útil de uma ferramenta deve ser calculado e considerado. Todas as ferramentas podem ter altos custos de manutenção e suporte.
- **Requisitos dos stakeholders:** É importante reunir os requisitos de todas os stakeholders para avaliar e identificar a ferramenta mais adequada. As ferramentas comerciais e as ferramentas de código aberto não necessariamente atendem a todos os requisitos em detalhes. As ferramentas personalizadas podem ser a melhor opção para atender a todos os requisitos individuais e nos casos em que nenhuma outra ferramenta oferece a funcionalidade necessária.

- **Cenário de software e estratégia de ferramenta existentes:** A composição existente de ferramentas (cenário de software) e a estratégia de ferramenta associada devem ser avaliadas, pois pode haver fornecedores preferenciais ou bloqueados, sistemas integrados que tenham dependências com outros produtos ou um modelo especial de suporte de serviço completo para todo o cenário de software com regulamentos específicos.

1.6.3 Considerações sobre o processo de seleção e avaliação do retorno sobre o investimento

As ferramentas de teste podem ser um investimento de longo prazo, talvez se estendendo por muitas iterações de um único projeto, e/ou aplicáveis a muitos projetos. Portanto, uma ferramenta em potencial deve ser considerada de diferentes pontos de vista.

- Para a gerência sênior, é necessário um ROI positivo;
- Para a equipe de suporte e operações, é preferível um número limitado, mas necessário, de ferramentas usadas em toda a organização. A manutenção de um número maior de ferramentas, o controle de suas licenças e o gerenciamento da pilha de ferramentas não devem consumir muito tempo nem custos;
- Para os líderes do projeto, a ferramenta deve agregar valor mensurável ao projeto ou à organização;
- Para as pessoas que usam a ferramenta, a usabilidade é muito importante. A usabilidade inclui, por exemplo, o suporte a determinadas tarefas, a capacidade de aprendizado e a operacionalidade;
- Para os membros da equipe operacional, a facilidade de manutenção é importante.

Os recursos devem ser analisados para cada tipo de ferramenta comercial e técnica. Diferentes perspectivas e interesses influenciam essa análise: gerenciamento de testes, análise de testes (técnicos), automação de testes ou desenvolvimento. A pessoa da organização responsável pela ferramenta (o proprietário da ferramenta) deve se certificar de que a análise seja realizada e que os pontos mencionados acima sejam considerados.

Todas as ferramentas introduzidas no processo de teste também devem garantir um ROI positivo para a organização. É responsabilidade do Gerente de Teste cuidar do cálculo e da avaliação adicional do ROI. No desenvolvimento ágil de software, isso pode ser responsabilidade de toda a equipe de desenvolvimento.

Deve-se realizar uma análise de custo-benefício antes de adquirir ou desenvolver uma ferramenta para garantir que ela beneficie a organização. Essa análise deve levar em conta os custos recorrentes e não recorrentes.

As atividades e os custos não recorrentes incluem o seguinte:

- Definir e determinar os requisitos da ferramenta para atender aos objetivos;
- Avaliação e seleção da ferramenta e do fornecedor corretos, prova de conceito;
- Adquirir, adaptar ou desenvolver a ferramenta para uso inicial;
- Definição de diretrizes para o uso da ferramenta;
- Treinamento inicial para a ferramenta;
- Integrar a ferramenta ao cenário de ferramentas existente;
- Aquisição do hardware/software necessário para dar suporte à ferramenta.

As atividades e os custos recorrentes incluem o seguinte:

- Taxas recorrentes de licenciamento e suporte;
- Custos de manutenção;
- Custos de treinamento contínuo;
- Portabilidade da ferramenta para diferentes ambientes.

Os custos de oportunidade também devem ser considerados. Isso significa que o tempo gasto na avaliação, administração, treinamento e uso da ferramenta poderia ter sido gasto em tarefas reais de teste. Portanto, podem ser necessários mais recursos de teste antes que a ferramenta possa ser usada para as atividades pretendidas.

Os seguintes riscos relacionados ao ROI devem ser considerados ao selecionar as ferramentas:

- A imaturidade da organização pode levar ao uso ineficiente da ferramenta;
- Mudanças na política de manutenção do fornecedor podem aumentar a carga de trabalho;
- Custos mais altos do que o esperado;
- Benefício menor do que o esperado.

Os benefícios a seguir podem se aplicar às ferramentas de teste:

- Redução do trabalho manual repetitivo (p. ex., testes de regressão);
- Aceleração do tempo de ciclo de teste por meio da automação;
- Redução dos custos de execução de testes por meio da diminuição das atividades manuais;
- Aumento da cobertura de determinados tipos de teste compatíveis com a ferramenta;
- Redução de erros humanos devido a menos atividades manuais;
- Acesso mais rápido às informações sobre os testes.

Benefícios e riscos adicionais, especialmente para ferramentas de automação de testes, podem ser encontrados no *ISTQB® Foundation Level Syllabus v4* e no *ISTQB® Test Automation Engineer Syllabus*.

Em geral, uma organização de testes raramente usa uma única ferramenta. O ROI total de uma organização costuma ser uma combinação do ROI de todas as ferramentas usadas para testes. As ferramentas precisam compartilhar informações e trabalhar de forma cooperativa. É recomendável ter uma estratégia abrangente e de longo prazo para as ferramentas de teste que inclua riscos, custos e benefícios.

1.6.4 Ciclo de vida da ferramenta

Há quatro estágios diferentes no ciclo de vida de uma ferramenta. Um administrador de ferramentas deve ser nomeado para garantir que as atividades desses estágios sejam definidas, executadas e gerenciadas.

- **Aquisição:** em primeiro lugar, foi tomada a decisão de selecionar uma ferramenta. Na segunda etapa, um proprietário da ferramenta precisa ser designado. Essa pessoa toma decisões sobre o uso da ferramenta (p. ex., convenções de nomenclatura de produtos de trabalho e onde esses produtos de trabalho serão armazenados). Tomar essas decisões antecipadamente pode fazer uma diferença significativa no eventual ROI da ferramenta.
- **Suporte e manutenção:** o proprietário da ferramenta é responsável pela manutenção da ferramenta. A responsabilidade pelas atividades de manutenção deve ser do administrador da ferramenta ou de um grupo dedicado de ferramentas. No caso da interoperabilidade, é preciso considerar o intercâmbio de dados e os processos de cooperação e comunicação. Além disso, são necessárias decisões sobre backup e restauração de artefatos relacionados à ferramenta.
- **Evolução:** Com o passar do tempo, o ambiente, as necessidades de negócio ou as decisões do fornecedor podem exigir alterações na ferramenta. Quanto mais complexo for o ambiente operacional de uma ferramenta, mais facilmente uma mudança poderá interromper seu uso.
- **Aposentadoria:** Ao final de sua vida útil, a ferramenta deve ser aposentada. Na maioria dos casos, a funcionalidade fornecida pela ferramenta será substituída e os dados deverão ser preservados e/ou arquivados. Isso pode ser feito com base em uma decisão do fornecedor ou porque se chegou a um ponto em que os benefícios e as oportunidades de mudar para uma nova ferramenta excedem seus custos e riscos.

1.6.5 Métricas de ferramentas

As métricas objetivas das ferramentas são projetadas e coletadas com base nas necessidades da equipe de teste e de outros stakeholders. As ferramentas de teste, em sua maioria, capturam dados valiosos em tempo real e reduzem os esforços de coleta de dados. Esses dados são usados para gerenciar o esforço geral de teste e identificar áreas para otimização.

Diferentes ferramentas se concentram na coleta de diferentes tipos de dados. Alguns exemplos são:

- As ferramentas de gerenciamento de testes podem fornecer uma variedade de métricas diferentes relacionadas a itens de teste disponíveis, testes, testes planejados, bem como status de execução de testes atuais e aprovados (p. ex., aprovados, reprovados, ignorados, bloqueados ou planejados).
- As ferramentas de gerenciamento de requisitos oferecem rastreabilidade em relação à cobertura dos requisitos por casos de teste aprovados e reprovados.
- As ferramentas de gerenciamento de defeitos podem fornecer informações sobre os defeitos, como status, gravidade, prioridade e densidade de defeitos dos itens de teste. Outros dados valiosos, como a porcentagem de detecção de defeitos, os níveis de teste nos quais os defeitos são introduzidos e o tempo de execução dos defeitos detectados ajudam a impulsionar o aprimoramento do processo, mas nem todos podem ser fornecidos apenas pela ferramenta de gerenciamento de defeitos.
- As ferramentas de análise estática, entre outras, fornecem métricas relacionadas à complexidade do código.
- As ferramentas de teste de performance podem fornecer informações valiosas, como tempos de resposta e taxas de falha sob cargas de pico.
- As ferramentas de cobertura de código ajudam a entender quais partes do objeto de teste foram exercitadas pelo teste.
- Embora as ferramentas de teste possam ser usadas para coletar métricas, elas também devem monitorar a si mesmas. Nesse contexto, a qualidade do processo de teste pode ser medida (p. ex., o número de defeitos encontrados com e sem ferramentas e a cobertura dos requisitos).
- Eficiência do teste (p. ex., duração da execução do teste e número de testes executados).

Mais detalhes sobre a coleta e o uso de métricas podem ser encontrados na Seção 2.1 deste syllabus, *Métricas de Teste*.

2 Gerenciando o Produto - 390 min

Palavras-chave

anomalia, defeito, relatório de defeito, fluxo de trabalho de defeitos, falha, métrica, estimativa de teste, objetivo do teste, progresso do teste

Palavras-chave específicas do domínio

planning poker, estimativa de três pontos, Wideband Delphi

Objetivos de aprendizagem:

2.1 Métricas de Teste

TM-2.1.1 (K2) Dar exemplos de métricas para atingir os objetivos do teste

TM-2.1.2 (K2) Explicar como controlar o progresso do teste usando métricas de teste

TM-2.1.3 (K4) Analisar os resultados dos testes para criar relatórios de testes que capacitem os stakeholders a tomar decisões

2.2 Estimativa de Teste

TM-2.2.1 (K2) Explicar os fatores que precisam ser considerados na estimativa de testes

TM-2.2.2 (K2) Dar exemplos de fatores que podem influenciar as estimativas de teste

TM-2.2.3 (K4) Selecionar uma técnica ou abordagem apropriada para a estimativa de testes para um determinado contexto

2.3 Gerenciamento de Defeitos

TM-2.3.1 (K3) Implementar um processo de gerenciamento de defeitos, incluindo o fluxo de trabalho de defeitos que possa ser usados para monitorar e controlar defeitos

TM-2.3.2 (K2) Explicar o processo e os participantes necessários para o gerenciamento eficaz de defeitos

TM-2.3.3 (K2) Explicar as especificidades do gerenciamento de defeitos no desenvolvimento ágil de software

TM-2.3.4 (K2) Explicar os desafios do gerenciamento de defeitos no desenvolvimento de software híbrido

TM-2.3.5 (K3) Usar os dados e as informações de classificação que devem ser coletados durante o gerenciamento de defeitos

TM-2.3.6 (K2) Explicar como as estatísticas de relatórios de defeitos podem ser usadas para planejar a melhoria do processo

2.1 Métricas de Teste

Introdução - Por que ter métricas de teste?

Há um ditado na administração que diz: "O que é medido, é feito". Da mesma forma, o que não é medido provavelmente não será feito porque é fácil de ignorar. Portanto, é importante estabelecer um conjunto adequado de métricas para qualquer empreendimento, inclusive os testes.

Os objetivos do teste são a resposta para a razão pela qual testamos (consulte a seção 1.4, *A Estratégia de Teste do Projeto*). Para determinar se os objetivos do teste foram atingidos, é preciso definir uma maneira de medi-los. As métricas de teste são os indicadores que nos ajudam a responder a essa pergunta.

As métricas de teste podem ser categorizadas da seguinte forma:

- **Métricas do projeto:** medir o progresso em relação aos critérios de saída existentes do projeto, como a porcentagem de testes executados, aprovados e reprovados.
- **Métricas do produto:** medir os atributos do produto, como o grau em que o produto atende às expectativas de qualidade pretendido pelos usuários.
- **Métricas de processo:** medir a capacidade do processo de teste e a eficácia do teste. Portanto, as métricas de processo são usadas para relatar a eficácia e a eficiência relacionadas ao processo.

Mais informações sobre o gerenciamento de métricas de produtos e processos podem ser encontradas no *ISTQB® Expert Test Management Syllabus*.

Mais informações sobre o uso de métricas de processo podem ser encontradas no *ISTQB® Expert Improving the Test Process Syllabus*.

As seções a seguir discutirão as métricas para planejamento de testes, monitoramento de testes, controle de testes e conclusão de testes. Essas são as quatro principais atividades de gerenciamento relacionadas às métricas.

2.1.1 Métricas para atividades de gerenciamento de testes

Este syllabus concentra-se nas seguintes atividades genéricas de gerenciamento de testes:

- Planejamento dos testes;
- Monitoramento e controle de testes;
- Conclusão do teste (consulte a seção 1.1, *O Processo de Teste*).

O gerenciamento de teste deve ser capaz de definir um conjunto de métricas para o monitoramento dos testes, controle dos testes e para a conclusão do teste como parte das atividades de planejamento de teste. Cada métrica precisa ser definida, medida, monitorada e relatada.

Durante o planejamento do teste, são definidas métricas apropriadas que correspondem aos objetivos de teste da estratégia de teste do projeto.

As métricas usadas durante o monitoramento e o controle do teste podem ser diferentes daquelas usadas na conclusão do teste. Durante o monitoramento e o controle do teste, as métricas são sobre o progresso das atividades do teste. Na conclusão do teste, os objetivos do teste devem ser alcançados. Uma ou mais métricas podem ser combinadas para medir os critérios de saída do teste atribuídos a determinados objetivos do teste.

A tabela a seguir fornece exemplos de métricas (há muitas outras) usadas nas atividades de gerenciamento de testes:

Métrica (planejada/monitorada para marcos definidos)	Monitoramento e Controle de Testes	Conclusão do Teste
Cobertura de requisitos	X	X
Cobertura dos riscos do produto	X	X
Cobertura de código	X	
Estimativa real vs. planejada (em horas) para atividades de teste	X	
Porcentagem de casos de teste executados por status (p. ex., com falha, bloqueados) vs. casos de teste planejados	X	X
Número acumulado de defeitos resolvidos vs. o número acumulado de defeitos	X	
Casos de teste automatizados reais vs. casos de teste automatizados planejados		X

Custo real vs. planejado dos testes	X	

Tabela 2: Exemplos de métricas usadas nas atividades de gerenciamento de testes

A métrica usada em uma atividade de teste específica é mostrada na tabela. As métricas com um X no monitoramento e no controle do teste são usadas principalmente para medir o progresso e são relatadas nos relatórios de progresso do teste (CTFL). As métricas com um X na conclusão do teste são usadas principalmente para medir a realização dos objetivos do teste e são relatadas no relatório de conclusão do teste (CTFL). As métricas com um X em ambas as colunas podem ser usadas para qualquer uma delas.

Há também métricas para monitorar a eficácia do teste (p. ex., porcentagem de detecção de defeitos (DDP)).

O DDP é abordado no syllabus do *ISTQB® Expert Level, módulo Improving the Test Process*, especificamente na seção *Implementing Test Process Improvement*.

2.1.2 Monitoramento, Controle e Conclusão

As métricas de teste são indicadores que mostram até que ponto o teste progrediu e se os critérios de saída ou as tarefas de teste relacionadas foram alcançados.

O monitoramento de testes é a atividade de coleta de dados referentes ao teste e à avaliação associada. Ele é usado para avaliar o progresso do teste e para verificar o cumprimento dos critérios de saída ou das atividades de teste associadas (consulte a seção 1.1.2, *Atividades de Monitoramento e Controle de Teste*). Os critérios de saída são derivados dos objetivos do teste.

O controle de testes usa as informações do monitoramento para fornecer orientação e ações corretivas para obter testes eficazes e eficientes. Exemplos de diretrizes de controle de teste incluem a redefinição de prioridades para os testes quando um risco identificado se torna um problema, a reavaliação se um item de teste atende aos critérios de entrada ou saída devido a retrabalho, o ajuste do cronograma de testes para levar em conta um atraso na entrega do ambiente de teste, e a adição de novos recursos quando e onde forem necessários.

A conclusão do teste coleta dados das atividades de teste concluídas para consolidar as lições aprendidas, o material de teste e outras informações relevantes. A conclusão do teste ocorre em marcos do projeto, como a conclusão de um nível de teste a conclusão de uma iteração, a conclusão (ou cancelamento) de um projeto de teste, o lançamento de um produto ou a conclusão de uma versão de manutenção.

As métricas de teste comuns usadas nas atividades de gerenciamento de teste incluem métricas de progresso do projeto e métricas que mostram o progresso em relação ao cronograma e ao orçamento planejados, a qualidade atual do item de teste e a eficácia do teste em relação aos objetivos do teste ou aos objetivos da iteração.

2.1.3 Relatório de Teste

O gerenciamento de testes deve entender como interpretar e usar as métricas para entender e relatar o status do teste. Para níveis mais altos de teste, como teste de sistema, teste de integração de sistema, teste de aceite e teste de segurança, a base de teste principal costuma ser produtos de trabalho, como especificações de requisitos, casos de uso, histórias de usuários e riscos do produto. As métricas de cobertura estrutural são mais aplicáveis a níveis mais baixos de teste, como teste de componentes (p. ex., cobertura de instrução) e teste de integração de componentes (p. ex., cobertura de interface). Embora o gerenciamento de testes possa usar métricas de cobertura de código para medir a extensão em que seus testes executam a estrutura do sistema em teste, o relatório dos resultados de testes de nível superior deve ser adaptado ao contexto e às necessidades específicas do projeto. Por exemplo, em ambientes que mudam com frequência, as métricas de cobertura de código podem ser úteis para monitorar o impacto das mudanças de código no conjunto de testes e identificar possíveis lacunas ou riscos. Além disso, o gerenciamento de testes deve entender que, mesmo que os testes de componentes e os testes de integração de componentes atinjam 100% de sua cobertura estrutural, os defeitos e os riscos de qualidade ainda precisam ser tratados em níveis de teste mais altos.

O objetivo do relatório de métricas é fornecer uma compreensão imediata das informações para fins gerenciais. As métricas podem ser relatadas como um instantâneo de uma métrica em um ponto no tempo ou como a evolução de uma métrica ao longo do tempo para avaliar tendências.

Riscos do produto, defeitos, progresso do teste, cobertura e custo relacionado, e o esforço de teste são medidos e relatados de maneiras específicas no final do projeto.

A seguir, exemplos de métricas que podem ser usadas para diferentes fins:

As **métricas relacionadas aos riscos** do produto incluem:

- Porcentagem de riscos em que todos os testes foram aprovados;
- Porcentagem de riscos em que alguns ou todos os testes foram reprovados;
- Porcentagem de riscos ainda não completamente testados.

Essas métricas podem ser usadas para avaliar a qualidade da base de teste e a eficácia dos casos de teste na cobertura dos riscos do produto.

As **métricas relacionadas a defeitos** incluem:

- Número acumulado de defeitos resolvidos versus o número acumulado de defeitos;
- Detalhamento do número ou da porcentagem de defeitos categorizados por:
  - Itens ou componentes de teste;
  - Origem do defeito (p. ex., especificação de requisitos, novo recurso ou regressão);
  - Versões de teste;
  - Nível de teste ou iteração introduzido, detectado e removido;
  - Prioridade/severidade;
  - Causa raiz;
  - Status (p. ex., rejeitado, duplicado, aberto, fechado).

Essas métricas podem ser usadas para monitorar o processo de detecção e resolução de defeitos, identificar as áreas de alta densidade ou gravidade de defeitos e avaliar a eficiência e a eficácia do teste.

As **métricas relacionadas ao progresso do teste** incluem:

- *Status da execução do teste*: Número total de testes planejados, implementados, executados, aprovados, reprovados, bloqueados e ignorados
- *Esforço de teste*: Número de horas de recursos reais versus planejadas dedicadas ao teste

As **métricas relacionadas à cobertura** incluem:

- *Cobertura de requisitos*: A porcentagem de requisitos que são cobertos por casos de teste;
- *Cobertura de risco do produto*: A porcentagem de riscos identificados do produto que são mitigados por casos de teste;
- *Cobertura de código*: A porcentagem de instruções de código, ramificações, caminhos ou condições que são executados pelos casos de teste.

As **métricas relacionadas aos custos e ao esforço de teste** incluem:

- *Riscos residuais para componentes não testados*: o impacto potencial e a probabilidade de defeitos nos componentes que não foram testados;
- *Custo do teste*: o custo real versus o custo planejado dos testes.

Além disso, é útil combinar métricas de diferentes categorias (p. ex., uma métrica que mostre a correlação entre as tendências de defeitos abertos e as tendências de testes executados, ou uma métrica que mostre a qualidade da base de testes com base no número de defeitos encontrados nos requisitos). Quando a execução do teste continua e cada vez menos defeitos são identificados, pode-se tomar a decisão de encerrar os testes. Essa decisão deve se basear no relatório das métricas e nos critérios de saída acordados.

2.2 Estimativas de Teste

Introdução

Existem práticas recomendadas de gerenciamento de projetos para a estimativa de engenharia de sistemas e software, com relação a todos os tipos de recursos (p. ex., custo, pessoas ou tempo). A estimativa de testes é a aplicação dessas práticas recomendadas aos testes associados a um projeto ou operação.

2.2.1 Estimativa das atividades que o teste envolverá

A estimativa de teste é uma atividade de gerenciamento de teste que calcula quanto tempo, esforço e custo uma tarefa levará para ser concluída. A estimativa de testes é uma das principais e mais importantes tarefas do gerenciamento de testes.

As principais características da estimativa no gerenciamento de testes são:

- **Esforço:** geralmente é calculado em homens/hora ou pontos de história necessários para concluir as tarefas de teste do projeto. Muitas vezes, o esforço de teste e a duração do teste (tempo decorrido) podem ser diferentes, e o gerenciamento de testes pode precisar estimar a duração total da atividade. Quantos homens/hora serão necessários?
- **Tempo:** necessário para concluir o projeto. O tempo é um recurso essencial em um projeto. O planejamento de testes precisa estimar o esforço de teste em dias corridos e em dias úteis. Todo projeto tem marcos e um prazo de entrega. Quanto tempo será necessário para concluir o projeto de teste?
- **Custo:** é o orçamento do projeto. Ele inclui as despesas com os recursos, as ferramentas e a infraestrutura de teste. Qual será o custo do projeto de teste?

Os testes geralmente são um subprojeto dentro de um projeto maior, às vezes distribuídos em vários locais de teste (p. ex., centros de teste). Para realizar a estimativa de testes a primeira etapa é identificar os níveis de teste, as atividades de teste e as tarefas de teste. Em seguida, divida o projeto de teste em suas principais atividades de teste (p. ex., planejamento de teste e execução de teste) dentro do processo de teste (consulte o *ISTQB® Foundation Level Syllabus v4*). Em projetos ágeis, as atividades de teste são estimadas com frequência dentro do trabalho de desenvolvimento, e não como valores separados. A próxima etapa é estimar o esforço de teste necessário para concluir as tarefas ou os produtos de trabalho e quais são os custos esperados com isso.

Como o teste é uma subatividade de um projeto, há sempre algumas restrições naturais do projeto que o influenciam e exigem compromisso, e não podemos manipular esses valores arbitrariamente. Isso pode ser encontrado no gerenciamento da qualidade como o triângulo tempo-custo-qualidade. No gerenciamento de projetos, o triângulo tempo-custo-qualidade compreende três valores que são interdependentes, o que significa que estão intimamente conectados e influenciam uns aos outros. Essa relação é comumente observada em cenários de projetos.

2.2.2 Fatores que podem influenciar o esforço de teste

A estimativa do esforço de teste envolve a previsão da quantidade de trabalho relacionado às atividades de teste que será necessária para atender aos objetivos de teste de um determinado projeto, versão ou iteração. Os fatores que influenciam o esforço de teste podem incluir as características do projeto:

Produto:

- A qualidade da base de teste;

- O tamanho do produto a ser testado (ou seja, o objeto de teste);
- A complexidade do domínio do produto (p. ex., ambiente, infraestrutura e histórico);
- Os requisitos para testar as características de qualidade (p. ex., segurança e confiabilidade).

Esses fatores relacionados ao produto podem influenciar as estimativas de teste porque criam um contexto específico para o sistema em teste.

Processo de desenvolvimento:

- A estabilidade e a maturidade dos processos de desenvolvimento da organização;
- O modelo de desenvolvimento (p. ex., desenvolvimento de software ágil/iterativo ou modelo híbrido de desenvolvimento de software) em uso;
- Os fatores materiais (p. ex., disponibilidade de automação de testes, ferramentas e ambientes de teste).

Esses fatores relacionados ao processo de desenvolvimento podem influenciar as estimativas de teste porque o teste está diretamente relacionado ao desenvolvimento.

Pessoas:

- Satisfação das pessoas (p. ex., proporcionada por feriados, férias, outros benefícios esperados);
- As habilidades e a experiência das pessoas envolvidas, especialmente em relação a projetos e produtos semelhantes (p. ex., conhecimento do domínio).

As pessoas são os recursos mais necessários, portanto, qualquer instabilidade deve ser levada em conta. Portanto, as pessoas são um fator importante na estimativa do esforço de teste. Consulte também a seção 3.1, *A Equipe de Teste*.

Resultados do teste:

- O número e a gravidade dos defeitos encontrados durante a execução do teste;
- A quantidade de retrabalho necessária.

Estimativa de teste de suporte de estatísticas históricas. Portanto, o conhecimento desses fatores ajudará a fazer uma estimativa com valores mais precisos.

Contexto do teste:

- A distribuição dos testes em várias subsidiárias, a composição e a localização das equipes, a complexidade do projeto (p. ex., vários subsistemas);
- O tipo de trabalho (p. ex., virtual ou no local).

Os fatores relacionados ao contexto são aqueles que afetam toda a estimativa do teste. Consulte também a Seção 1.2, *O Contexto dos Testes*.

2.2.3 Seleção de técnicas de estimativa de teste

A estimativa de teste deve abranger todas as atividades envolvidas no processo de teste. O custo, o esforço e, principalmente, a duração estimados da execução do teste costumam ser os mais importantes para o gerenciamento de testes, pois esses valores afetarão o projeto. Entretanto, as estimativas de execução de teste tendem a ser difíceis de gerar quando a qualidade geral do software é baixa ou desconhecida. Além disso, a familiaridade e a experiência com o produto provavelmente afetarão a qualidade das estimativas. Uma prática comum é estimar o número de casos de teste derivados da base de teste (p. ex., requisitos ou histórias de usuários). As suposições feitas durante a estimativa de teste devem ser sempre documentadas como parte da estimativa.

As técnicas ou abordagens de estimativa de teste podem ser categorizadas como baseadas em métricas ou baseadas em especialistas.

Mais detalhes sobre a estimativa de teste As técnicas são explicadas no *ISTQB® Foundation Level Syllabus v4*.

Na maioria dos casos, a estimativa, uma vez preparada, deve ser entregue ao gerenciamento do projeto, juntamente com uma justificativa. Frequentemente, alguns parâmetros de entrada são alterados (p. ex., o escopo do teste), resultando, muitas vezes, no ajuste da estimativa. O ideal é que a estimativa final do teste represente o melhor equilíbrio possível entre as metas organizacionais e do projeto nas áreas de qualidade, cronograma, orçamento e recursos.

É importante ter em mente que qualquer estimativa é baseada nas informações disponíveis quando é preparada. No início de um projeto, as informações podem ser bastante limitadas. Além disso, essas informações podem mudar com o tempo. Para permanecerem precisas, as estimativas devem ser atualizadas para refletir as informações novas e em constante mudança.

A seleção da técnica de estimativa depende de vários fatores, como:

- **Erro de estimativa:** algumas técnicas fornecem uma maneira de calcular o desvio padrão, que é uma medida da incerteza ou variabilidade da estimativa. Por exemplo, a técnica de estimativa de três pontos usa as estimativas otimista, pessimista e mais provável para calcular o valor esperado e o desvio padrão da estimativa (consulte o *ISTQB® Foundation Level Syllabus v4*, seção 5.1.4 para obter mais detalhes).
- **Disponibilidade de dados:** algumas técnicas exigem dados históricos de projetos anteriores ou semelhantes que podem não estar disponíveis ou ser confiáveis. Por exemplo, a estimativa baseada em proporções e extrapolação depende de dados históricos para derivar as proporções ou as tendências do projeto atual.
- **Disponibilidade de especialistas:** algumas técnicas exigem o envolvimento de especialistas que tenham o conhecimento e a experiência necessários para fornecer estimativas precisas e realistas. Por exemplo, o *Wideband Delphi* e o *planning poker* dependem das opiniões e julgamentos de especialistas ou membros da equipe.
- **Conhecimento em modelagem:** algumas técnicas exigem o uso de modelos matemáticos ou fórmulas para calcular as estimativas, o que pode exigir algumas habilidades e conhecimentos em modelagem. Por exemplo, a extrapolação e a estimativa de três pontos usam fórmulas para derivar o valor esperado e o desvio padrão da estimativa.
- **Restrições de tempo:** algumas técnicas exigem mais tempo e esforço para serem executadas do que outras, o que pode afetar a viabilidade e a adequação da técnica. Por exemplo, o *planning poker* é fácil de ser feito, enquanto a extrapolação pode ser mais difícil.

Isso mostra que os critérios de seleção das técnicas adequadas de estimativa de teste são altamente dependentes do contexto do teste (p. ex., SDLC, stakeholders, níveis de teste e tipos de teste usados no projeto) (consulte a seção 1.2, *O Contexto do Teste*). O Gerente de Teste deve ser capaz de coordenar e aplicar as técnicas de estimativa de testes (p. ex., com diferentes modelos de SDLC em um projeto em diferentes subsidiárias).

Por exemplo, para selecionar a técnica de estimativa adequada, primeiro determine a complexidade do tópico. Se a complexidade for baixa, poderão ser usadas técnicas baseadas em métricas. Se a complexidade for alta, poderão ser usadas técnicas baseadas em especialistas. Se for usado um modelo de desenvolvimento sequencial, a técnica de estimativa *Wideband Delphi* poderá ser usada. Se for usado um modelo de desenvolvimento de software ágil, poderá ser usado o *planning poker*.

2.3 Gerenciamento de Defeitos

Introdução

O *ISTQB® Foundation Level Syllabus v4* descreve as atividades que começam após a observação de resultados reais que diferem dos resultados esperados. O syllabus se refere a essas atividades como gerenciamento de defeitos. Outros padrões usam o termo "gerenciamento de incidentes", padrão ISO/IEC/IEEE 29119-3, ou "gerenciamento de anomalias" (TMAP) para enfatizar o fato de que, no início do processo, talvez não saibamos se a discrepância é

causada por um defeito em um produto de trabalho ou por outra coisa (p. ex., falha na automação de testes ou um mal-entendido dos requisitos pelo testador). O gerenciamento de defeitos e a ferramenta usada para gerenciá-los são de importância fundamental para os testadores e para outros membros da equipe envolvidos no desenvolvimento de software. As informações de um processo eficaz de gerenciamento de defeitos permitem que a equipe de teste e outros stakeholders do projeto tenham uma visão do estado de um projeto em todo o seu SDLC. O gerenciamento de defeitos também é fundamental para decidir quais defeitos serão corrigidos. Isso garante que o esforço seja gasto no trabalho com os defeitos corretos. A coleta e a análise de dados relacionados a defeitos ao longo do tempo podem ajudar a localizar áreas de possível melhoria tanto para testes quanto para outros processos dentro do SDLC (p. ex., melhor prevenção de defeitos por meio de arquitetura e design técnico aprimorados). por meio de arquitetura e projeto técnico aprimorados).

Além de compreender o ciclo de vida geral dos defeitos e como ele é usado para monitorar e controlar os processos de desenvolvimento e teste de software, o Gerente de Teste e os testadores (ou toda a equipe ágil no desenvolvimento ágil de software) também devem estar familiarizados com os dados críticos a serem capturados. O Gerente de Teste deve ser um defensor do uso adequado do processo de gerenciamento de defeitos e da ferramenta de gerenciamento de defeitos selecionada.

2.3.1 Ciclo de Vida do Defeito

Cada fase do SDLC deve incluir atividades para detectar e remover possíveis defeitos. Por exemplo, técnicas de teste estático (ou seja, revisões e análise estática) podem ser usadas em especificações de projeto, especificações de requisitos e código antes de entregar esses produtos de trabalho em atividades subsequentes. Quanto mais cedo cada defeito for detectado e removido, menor será o custo geral da qualidade do produto. do produto. O custo da qualidade é minimizado quando cada defeito é removido na mesma fase em que foi introduzido (ou seja, quando o processo de software atinge a contenção perfeita de fases).

Durante o teste estático, procuramos por defeitos. Durante o teste dinâmico, a presença de um defeito é revelada quando ele causa uma falha que resulta em uma discrepância entre os resultados reais e os resultados esperados de um teste (ou seja, uma anomalia). Em alguns casos, um resultado falso-negativo ocorre quando o testador não observa a anomalia. Quando uma anomalia é observada, deve ser realizada uma investigação mais aprofundada. Essa investigação geralmente começa com o preenchimento de um relatório de defeitos de acordo com o processo definido de gerenciamento de testes e defeitos. Um teste com falha nem sempre resulta na criação de um relatório de defeitos. (p. ex., no desenvolvimento orientado por testes, em que os testes de componentes, geralmente automatizados, são usados como uma forma de especificação de projeto executável). Até que o desenvolvimento do componente seja concluído, alguns ou todos os testes devem falhar inicialmente. Portanto, o resultado desse teste não é necessariamente causado por um defeito e, normalmente, não é rastreado por meio de um relatório de defeitos.

Um relatório de defeito progride ao longo de um fluxo de trabalho (para simplificar e manter a consistência com a maioria das ferramentas de gerenciamento de defeitos, continuaremos a usar o termo "fluxo de trabalho de defeitos") e passa por uma sequência de estados de defeitos. Na maioria desses estados, uma pessoa possui o relatório de defeitos e é responsável pela execução de uma tarefa (p. ex., análise, remoção de defeitos ou teste de confirmação). O diagrama a seguir representa um fluxo de trabalho de defeito simples:

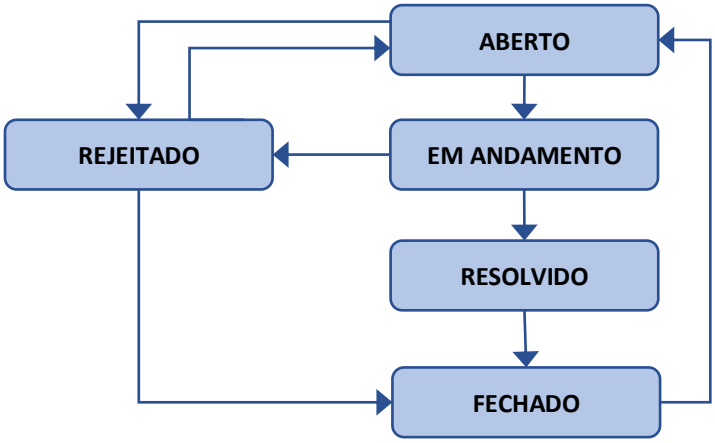

Figura 2: Um fluxo de trabalho simples sobre defeitos

Um fluxo de trabalho simples de defeitos pode abranger os seguintes estados de defeito:

- **ABERTO** (pode ser chamado de NOVO): O estado inicial quando o relatório de defeitos é criado.
- **EM ANDAMENTO**: A equipe está trabalhando no relatório de defeitos, análise e/ou correção.
- **REJEITADO**: Um relatório de defeito é rejeitado pela pessoa que o processou (geralmente um desenvolvedor ou um analista). Pode haver muitos motivos para a rejeição (p. ex., informações inválidas, teste incorreto, relatório de defeitos duplicado) e essas informações são adicionadas ao relatório de defeitos.
- **RESOLVIDO** (pode ser chamado de CORRIGIDO, PRONTO PARA RETESTE): Um testador executa um teste de confirmação, geralmente seguindo as etapas para reproduzir a falha do relatório de defeitos para determinar se a correção de fato resolveu o defeito.
- **FECHADO**: O relatório de defeitos atingiu seu estado terminal e não se pretende realizar mais nenhum trabalho. O testador faz a transição do relatório de defeitos para esse estado após um teste de confirmação bem-sucedido ou para reconhecer a rejeição do relatório de defeitos.

Um fluxo de trabalho de defeito simples é usado em muitas organizações e é ampliado pelo uso de outros estados de defeito relevantes para um determinado contexto (p. ex., REABERTO, ACEITO, DETALHAMENTO ou DEFERIDO).

O fluxo de trabalho de defeitos pode variar em diferentes organizações em termos de nomes diferentes para os estados de defeito, regras para transições entre estados de defeito e funções responsáveis por tarefas em determinados estados de defeito. Em geral, o fluxo de trabalho de defeitos é mais simples no desenvolvimento ágil de software do que nos modelos de desenvolvimento sequencial. O fluxo de trabalho de defeitos deve ser adaptado a um determinado contexto. Ao projetar o fluxo de trabalho de defeitos, é aconselhável respeitar várias boas práticas:

- Se possível, o fluxo de trabalho de defeitos deve ser definido em toda a organização para proporcionar um gerenciamento unificado em todos os projetos;
- Defeitos duplicados e falso-positivos devem ser representados por um estado separado ou por uma combinação do status REJEITADO com a escolha do motivo da rejeição. Eles podem ser úteis em outras análises de defeitos com o objetivo de melhorar o processo de teste;
- Recomenda-se o uso de apenas um estado terminal (p. ex., FECHADO). A transição para esse estado geralmente requer a escolha de um motivo para o fechamento, útil para a avaliação do processo e para as atividades de melhoria do processo;
- Os nomes dos estados no fluxo de trabalho de defeitos devem ser os mesmos dos estados análogos usados para outras entidades (p. ex., histórias de usuários e tarefas de teste) para simplificar o trabalho com eles;

- Os estados de defeito consecutivos devem pertencer a funções responsáveis diferentes. Se dois ou mais estados consecutivos pertencerem à mesma função responsável, deve haver um bom motivo (p. ex., para medir o tempo gasto em um estado de defeito);
- Cada estado de defeito, exceto o estado terminal, deve ter mais de uma transição de saída para permitir que a função responsável tome uma decisão sobre a próxima etapa. As exceções a essa regra devem ser justificadas (p. ex., para monitorar o tempo gasto em uma determinada atividade);
- O conjunto de atributos que devem ser inseridos ao se realizar uma transição de estado deve ser limitado àqueles que dão valor substancial ao gerenciamento de defeitos.

2.3.2 Gerenciamento de defeitos multifuncional

Embora a organização de testes e o Gerente de Teste geralmente sejam proprietários do processo geral de gerenciamento de defeitos e da ferramenta de gerenciamento de defeitos, uma equipe multifuncional geralmente é responsável pelo gerenciamento dos defeitos de um determinado projeto. Essa equipe, às vezes chamada de comitê de gerenciamento de defeitos, pode incluir o Gerente de Teste, representantes do desenvolvimento, fornecedores, gerenciamento de projetos, gerenciamento de produtos ou Product Owner e outros stakeholders que tenham interesse no software em teste.

À medida que as anomalias são descobertas e inseridas na ferramenta de gerenciamento de defeitos, o comitê de gerenciamento de defeitos deve determinar se cada relatório de defeito representa um defeito válido e se ele deve ser corrigido (e por quem, caso várias equipes de desenvolvimento estejam participando da entrega), rejeitado ou adiado. Essa decisão exige que o comitê de gerenciamento de defeitos considere os benefícios, os riscos e os custos associados à correção do defeito. É vantajoso discutir a consideração em uma reunião (geralmente chamada de reunião de triagem). Se o defeito tiver de ser corrigido, a equipe deverá estabelecer a prioridade da correção do defeito em relação a outras tarefas. O Gerente de Teste e a equipe de testes podem ser consultados sobre a importância relativa de um defeito e devem fornecer as informações objetivas disponíveis.

Em projetos muito grandes, a nomeação de um gerente de defeitos em tempo integral pode ser justificada pelo esforço necessário para preparar e acompanhar as decisões tomadas nas reuniões do comitê de gerenciamento de defeitos, pelo menos durante as fases do SDLC, quando os testes são mais intensos. Em outras situações, vários projetos grandes podem compartilhar um gerente de defeitos.

Uma ferramenta de gerenciamento de defeitos não deve ser usada como substituto para uma boa comunicação, nem um comitê de gerenciamento de defeitos deve ser usado como substituto para o uso eficaz de uma boa ferramenta de gerenciamento de defeitos. A comunicação, o suporte adequado à ferramenta, um fluxo de trabalho de defeitos bem definido (incluindo as propriedades do relatório de defeitos) e uma equipe de gerenciamento de defeitos engajada são todos necessários para um gerenciamento de defeitos eficaz e eficiente.

2.3.3 Especificidades do gerenciamento de defeitos em equipes ágeis

O gerenciamento de defeitos em organizações que usam o desenvolvimento ágil de software geralmente é mais leve e/ou menos formal do que nos modelos de desenvolvimento sequencial. Se as equipes ágeis estiverem colocalizadas ou tiverem disponíveis meios de comunicação bem estabelecidos, as informações sobre um defeito ou falha serão frequentemente trocadas entre testadores, representantes de clientes e desenvolvedores sem um relatório formal de defeitos. No entanto, os relatórios de defeitos devem ser criados para:

- Defeitos que bloqueiam outras atividades atuais do sprint (ou seja, desenvolvimento, testes ou outros) e que não podem ser corrigidos imediatamente na equipe Ágil.
- Defeitos que não podem ser resolvidos na mesma iteração. Algumas equipes ágeis têm a regra de criar um relatório de defeitos se o defeito não puder ser resolvido durante o dia em que a falha for encontrada.
- Defeitos que devem ser resolvidos por ou em cooperação com outras equipes em organizações com várias equipes.
- Defeitos que devem ser resolvidos por um fornecedor.

- Defeitos em que um relatório de defeitos é explicitamente solicitado (p. ex., quando um desenvolvedor não pode trabalhar imediatamente em uma correção).

A prática comum é adicionar defeitos que não podem ser resolvidos na mesma iteração ao backlog do produto para que possam ser priorizados entre outros defeitos e histórias de usuários para uma iteração posterior.

Embora os fundamentos do gerenciamento de defeitos devam ser definidos na estratégia de teste da organização, muitos aspectos, incluindo o nível de formalidade, os gatilhos para a criação de um relatório de defeitos, podem ser considerados. Muitos aspectos, inclusive o nível de formalidade, os gatilhos para a criação de um relatório de defeitos e os atributos de defeitos a serem capturados podem ser deixados para o acordo entre os membros da equipe Ágil. Em geral, o nível de formalidade do gerenciamento de defeitos e a abordagem para a criação de relatórios de defeitos devem refletir o seguinte:

- Co-localização dos membros da equipe;
- Distribuição dos membros da equipe entre fusos horários;
- O número de equipes que cooperam no desenvolvimento de produtos;
- Maturidade da(s) equipe(s);
- Tamanho da(s) equipe(s);
- Riscos associados ao produto;
- Requisitos regulatórios, contratuais ou outros (se é onde aplicável).

A decisão final da equipe Ágil sobre os detalhes do gerenciamento de defeitos deve ser sempre documentada (p. ex., com diretrizes em uma ferramenta de gerenciamento de conhecimento).

2.3.4 Desafios do gerenciamento de defeitos no desenvolvimento de software híbrido

Na prática, várias equipes geralmente colaboram na entrega do sistema ou do sistema de sistemas. Os exemplos incluem o desenvolvimento de software híbrido quando um cliente usa o desenvolvimento ágil de software e um de seus fornecedores usa um modelo de desenvolvimento sequencial ou quando uma organização que usa um modelo de desenvolvimento sequencial exige a entrega de um subsistema de uma equipe que usa o desenvolvimento ágil de software. Esse ambiente com várias equipes apresenta vários desafios:

- **Alinhamento dos atributos de defeitos e das ferramentas a serem usadas para o gerenciamento de defeitos:** Em um cenário ideal, todas as equipes usam uma ferramenta de gerenciamento de defeitos. Na prática, é comum que cada equipe use uma ferramenta de gerenciamento de defeitos diferente, especialmente quando várias equipes de fornecedores contribuem para a entrega do projeto. Nesses casos, é bom estabelecer a sincronização entre as ferramentas de gerenciamento de defeitos (de preferência, automaticamente).
- **Priorização de defeitos:** O(s) Product Owner(s) deve(m) ser incluído(s) nas reuniões de gerenciamento de defeitos e buscar ativamente informações sobre as consequências e os riscos associados aos defeitos. As reuniões de gerenciamento de defeitos devem ser realizadas com mais frequência no desenvolvimento ágil de software do que nos modelos de desenvolvimento sequencial para acompanhar o ritmo mais rápido de entrega de incrementos de produtos da equipe ágil. Essas reuniões podem, no entanto, ser mais curtas com equipes ágeis. Às vezes, é vantajoso que um grupo menor de stakeholders no gerenciamento de defeitos tenha a palavra final sobre a priorização dos defeitos.
- **Alinhamento e transparência do plano de teste para novos desenvolvimentos e correções de defeitos**: O trabalho de todas as equipes deve estar alinhado ao mesmo plano de projeto, independentemente de estarem usando o desenvolvimento ágil de software ou modelos de desenvolvimento sequencial. Todas as entregas, inclusive as correções de defeitos, devem estar alinhadas com esse plano de projeto. É possível obter um melhor alinhamento com a participação ativa dos membros de todas as equipes no processo de planejamento (p. ex., a participação das equipes do modelo de desenvolvimento sequencial em reuniões de desenvolvimento de software ágil em que os defeitos são discutidos e priorizados). A transparência dos

planos de desenvolvimento pode ser melhorada compartilhando-os entre as equipes (p. ex., por meio de painéis ou do backlog do produto).

2.3.5 Informações do relatório de defeitos

As informações em um relatório de defeitos devem ser suficientes para as seguintes finalidades:

- Gerenciamento do relatório de defeitos durante o ciclo de vida do defeito;
- Avaliação do status geral do projeto, especialmente em termos de qualidade do produto e progresso dos testes;
- Avaliação do status de um incremento de produto em termos de qualidade do produto;
- Avaliação da capacidade do processo.

As informações necessárias para o gerenciamento de defeitos e o status do projeto podem variar dependendo de quando o defeito é detectado no SDLC. Além disso, os relatórios de defeitos relacionados a características de qualidade não funcionais podem precisar de mais informações (p. ex., condições de carga para problemas de performance). Entretanto, as principais informações coletadas devem ser consistentes em todo o SDLC e, preferencialmente, em todos os projetos de uma organização para permitir uma comparação significativa dos dados de defeitos em todo o projeto e em todos os projetos.

Muitos itens de dados podem ser coletados em um relatório de defeitos. O Gerente de Teste deve decidir quais informações são apropriadas para o gerenciamento eficaz de defeitos em um determinado contexto de projeto. Devido ao fato de que cada atributo adicional aumenta o tempo gasto no relatório de defeitos e pode aumentar a confusão da pessoa que está inserindo o relatório de defeitos, é aconselhável coletar apenas os dados necessários para o gerenciamento de defeitos em um determinado contexto e/ou que serão usados para a melhoria do processo.

Para gerenciar o relatório de defeitos na maioria dos ambientes, os itens a seguir são obrigatórios:

- Um título de defeito com um breve resumo da anomalia;
- Uma descrição detalhada da anomalia, de preferência incluindo as etapas para reproduzir a falha;
- Gravidade do impacto no sistema em teste e/ou nos stakeholders do produto;
- Prioridade para corrigir a anomalia.

Itens de dados adicionais importantes geralmente são criados pela ferramenta de gerenciamento de defeitos:

- Identificador exclusivo para o relatório de defeitos;
- Data/hora da criação do relatório de defeitos;
- Nome da pessoa que descobriu e/ou relatou a anomalia;
- Fase do projeto e do SDLC em que a anomalia foi descoberta;
- Estado atual do relatório de defeitos;
- Proprietário atual (ou seja, a pessoa atualmente designada para trabalhar no defeito);
- Histórico de mudanças, como a sequência de ações, incluindo informações de data/hora, tomadas pelos membros da equipe do projeto para isolar, reparar e confirmar o defeito como corrigido;
- Referências (p. ex., ao caso de teste, a defeitos conectados).

Dependendo do contexto, outras informações (p. ex., rastreabilidade) também podem ser coletadas em um relatório de defeitos (consulte a ISO/IEC/IEEE 29119-3 para obter mais informações). Os itens a seguir agrupam as informações de acordo com a finalidade pretendida:

- **Para ajudar na resolução do defeito**: O subsistema ou componente no qual o defeito se encontra, o item de teste específico e seu número de versão no qual a anomalia foi observada ou o ambiente de teste no qual o defeito foi observado.
- **Para avaliar o status geral do projeto:** Informações para monitorar o progresso (p. ex., riscos, custos, oportunidades e benefícios associados à correção ou não do defeito, uma descrição de qualquer solução alternativa disponível ou requisitos afetados pelos defeitos).
- **Para avaliar o status de um incremento de produto em termos de qualidade do produto:** O tipo de defeito (geralmente correspondente a uma taxonomia de defeitos), o produto de trabalho no qual o defeito foi introduzido ou a característica/subcaracterística de qualidade afetada pelo defeito.
- **Para avaliar a capacidade do processo**: Informações para monitorar a eficácia e a eficiência dos processos de desenvolvimento (p. ex., a fase do SDLC de introdução, detecção e remoção do defeito ou da causa raiz do defeito).

2.3.6 Definição de ações de melhoria do processo usando informações do relatório de defeitos

Conforme discutido na seção 2.3.5 deste syllabus, Informações sobre o relatório de defeitos, os relatórios de defeitos podem ser úteis para o monitoramento e o relatório do status do projeto. Embora as implicações das métricas no processo de teste sejam abordadas principalmente no *ISTQB Expert Test Management Syllabus*, no nível *Advanced Test Management*, os gerentes de teste devem estar cientes do que os relatórios de defeitos significam para avaliar a capacidade dos processos de desenvolvimento e teste de software.

Além das informações de monitoramento do progresso do teste mencionadas neste syllabus, na seção 2.1.2, *Monitoramento, Controle e Conclusão*, e na seção 2.1.3, *Relatório de Testes*, as informações sobre defeitos devem apoiar as iniciativas de melhoria do processo, conforme discutido durante as retrospectivas. Os exemplos incluem:

- Usar informações sobre as fases de introdução, detecção e remoção de defeitos para avaliar a contenção de fases e/ou realizar análises de custo de qualidade com o objetivo de sugerir maneiras de melhorar a eficácia da detecção de defeitos em cada fase e minimizar o custo associado a defeitos. com o objetivo de sugerir maneiras de melhorar a eficácia da detecção de defeitos em cada fase e minimizar o custo associado aos defeitos.
- Uso de informações sobre a fase de introdução para análise das fases nas quais o maior número de defeitos é introduzido, para permitir melhorias direcionadas à prevenção de defeitos.
- Uso de informações sobre a causa-raiz dos defeitos informações para determinar os motivos subjacentes à introdução de defeitos, para permitir melhorias no processo que reduzam o número total de defeitos.
- Usar informações de localização de defeitos para realizar análises de grupos de defeitos, para entender melhor os riscos técnicos (para testes baseados em riscos) e para permitir a refatoração de componentes problemáticos.
- Usar informações sobre defeitos reabertos para avaliar a qualidade das implementações de depuração.
- Usar informações sobre defeitos duplicados e rejeitados para avaliar a qualidade da criação do relatório de defeitos.
- Permitir melhorias no processo que reduzam o número total de defeitos por meio da introdução de medidas proativas para evitar erros antecipadamente.

O uso de métricas para avaliar a eficácia e a eficiência do processo de teste é discutido no *ISTQB Expert Test Management Syllabus*.

Em alguns casos, as equipes decidem não rastrear os defeitos encontrados durante algumas ou todas as fases do SDLC. Embora isso seja feito com frequência em nome da eficiência e com o objetivo de reduzir a sobrecarga do processo, reduz bastante a visibilidade dos recursos do processo de desenvolvimento e teste de software. Isso dificulta a realização das melhorias sugeridas acima devido à falta de dados de suporte confiáveis.

3 Gerenciando a Equipe - 225 min

Palavras-chave

avaliação, custo da qualidade, defeito, prevenção de defeitos, falha externa, falha interna

Objetivos de aprendizagem:

3.1 A Equipe de Teste

TM-3.1.1 (K2) Dê exemplos de habilidades típicas necessárias para os membros da equipe de teste em quatro áreas de competência

TM-3.1.2 (K4) Analisar um determinado contexto de projeto para determinar as habilidades necessárias para os membros da equipe de teste

TM-3.1.3 (K2) Explicar as técnicas típicas de avaliação de habilidades dos membros da equipe de teste

TM-3.1.4 (K2) Diferencie as abordagens típicas para desenvolver as habilidades dos membros da equipe de teste

TM-3.1.5 (K2) Explicar as habilidades necessárias para gerenciar uma equipe de teste

TM-3.1.6 (K2) Dê exemplos de fatores de motivação e higiene para os membros da equipe de teste

3.2 Relacionamento com os Stakeholders

TM-3.2.1 (K2) Dê exemplos de cada uma das quatro categorias que determinam o custo da qualidade

TM-3.2.2 (K3) Aplicar um cálculo de custo-benefício para estimar o valor agregado dos testes para os stakeholders

3.1 A Equipe de Teste

Introdução

Qualquer equipe que execute tarefas de teste é formada por indivíduos com diferentes competências. Enquanto em algumas organizações as equipes são auto-organizadas, em outras, os gerentes de teste recrutam e desenvolvem essas equipes. A combinação certa de habilidades[1] é um fator essencial para que todas as equipes concluam com êxito as tarefas de teste.

As habilidades exigidas por um membro da equipe de teste podem mudar com o tempo. É importante selecionar as pessoas certas para a equipe de teste e oferecer treinamento adequado e oportunidades de crescimento. Além disso, pessoas de fora da equipe de teste podem fornecer habilidades específicas adicionais.

Esta seção aborda o processo fundamental de análise e desenvolvimento das habilidades necessárias dos membros da equipe de teste, bem como as habilidades necessárias para liderar ou treinar uma equipe de teste. Isso também inclui o conhecimento dos fatores que motivam ou desmotivam os membros da equipe de teste e outros fatores para garantir o sucesso do trabalho em equipe.

Cada indivíduo já tem habilidades e pode desenvolvê-las ainda mais de várias maneiras, como experiência de trabalho, educação e treinamento. A equipe de teste ideal tem todas as habilidades necessárias para determinadas tarefas de teste ou é responsável apenas pelas tarefas para as quais tem as habilidades necessárias. Para ser bem-sucedida, uma equipe de teste precisa de várias habilidades em diferentes níveis. Dependendo do contexto do projeto, algumas habilidades serão mais importantes ou necessárias do que outras. Pode fazer sentido trazer especialistas externos para tarefas de teste específicas que estejam além das capacidades da equipe de teste.

3.1.1 Habilidades típicas em quatro áreas de competência

As habilidades de uma pessoa podem ser classificadas em quatro áreas de (Sonntag & Schmidt-Rathjens, 2005) (Erpenbeck & von Rosenstiel, 2017) [2] :

- **Competência Profissional:** consiste em habilidades para realizar tarefas especializadas. Os exemplos incluem habilidades em técnicas de teste, conhecimento tecnológico e de negócio no domínio do aplicativo, bem como habilidades de gerenciamento de projetos.
- **Competência Metodológica:** inclui habilidades gerais que uma pessoa pode usar de forma independente em um domínio e que permitem a performance independente de tarefas complexas ou novas. Os exemplos incluem habilidades analíticas, conceituais e de julgamento.
- **Competência Social:** inclui habilidades relacionadas à comunicação, à cooperação e ao gerenciamento de conflitos em contextos intra e interculturais. Elas permitem que a pessoa se relacione com os outros para agir adequadamente em uma determinada situação e atingir metas individuais e compartilhadas. Os exemplos incluem habilidades de comunicação, habilidades de resolução de conflitos, capacidade de trabalhar em equipe, adaptabilidade e assertividade.
- **Competência Pessoal:** inclui a capacidade e a vontade de se desenvolver e de desenvolver o próprio talento, a motivação e a vontade de realizar, bem como o desenvolvimento de atitudes específicas e de uma personalidade individual. Os exemplos incluem autogerenciamento, responsabilidade pessoal, capacidade de

[1] O termo "habilidade" é usado como um termo abrangente para as próprias habilidades, para ter conhecimento de algo e para ter a capacidade de fazer algo.

[2] As quatro áreas de competência usadas aqui são baseadas no modelo descrito nessas referências, que é amplamente utilizado. Há outros modelos descritos na literatura que agrupam as habilidades de forma diferente. Eles não fazem parte deste programa de estudos

receber críticas, confiabilidade, resiliência, capacidade de agir com confiança, disciplina, abertura a mudanças, vontade de ajudar e aprender e capacidade de delegar.

Todas as áreas de competência são importantes para o sucesso de qualquer equipe de testes. Como a competência metodológica, social e pessoal não é específica dos testes, o ISTQB® se concentra no desenvolvimento da competência profissional. Isso inclui habilidades para gerenciar tarefas de teste, analisar a base de teste, modelar testes, identificar e analisar riscos e desenvolver, configurar e manter dados de teste, ambientes de teste e scripts de teste.

3.1.2 Analisar as habilidades necessárias dos membros da equipe de teste

A alocação de pessoal é uma atividade do planejamento de testes. Isso inclui a tarefa de identificar as funções e as habilidades da equipe necessárias para implementar os testes na estratégia de teste. É necessária uma análise detalhada do contexto para determinar as habilidades necessárias para um projeto.

Competência profissional e metodológica

Para os testes, o foco está nas habilidades necessárias para as tarefas de teste. Veja a seguir alguns exemplos:

- O planejamento de testes requer conhecimento conceitual para desenvolver uma estratégia de teste.
- Monitoramento e controle de testes exigem habilidades de gerenciamento de projetos para gerenciar todas as tarefas de teste.
- A análise de teste requer habilidades analíticas para analisar a base do teste e os riscos do produto.
- O projeto de teste requer habilidades em técnicas de teste para projetar casos de teste e conhecimento conceitual para projetar os ambientes de teste.
- A implementação de testes requer habilidades de julgamento para a seleção de testes e conhecimento técnico para a programação de scripts de teste e a configuração de ambientes de teste.
- A execução de testes requer conhecimento técnico para executar testes automatizados, realizar testes exploratórios e avaliar os resultados dos testes.
- A conclusão do teste requer a capacidade de comunicar os resultados do projeto e a responsabilidade pessoal pelas decisões tomadas.

Diferentes tipos e níveis de teste exigem diferentes habilidades (p. ex., conhecimento comercial no domínio do aplicativo para avaliar a adequação funcional de um sistema, ou conhecimento técnico para avaliar a capacidade de manutenção do código).

Além disso, o contexto do projeto fornece informações valiosas sobre a competência profissional necessária:

- O domínio do sistema requer experiência no negócio na área de aplicativos, como tecnologia da informação, setor automotivo ou setor de jogos de azar.
- A arquitetura e as tecnologias de software e sistema usadas no projeto exigem, por exemplo, conhecimento técnico especializado em linguagens de programação, tecnologia de interface ou vulnerabilidades de segurança.
- O SDLC requer, por exemplo, conhecimento sobre níveis de teste, funções de teste e técnicas de teste específicas.

Competência Social

No contexto do teste, a competência social permite que os membros da equipe de teste se comportem adequadamente no relacionamento com outros membros da equipe e atinjam os objetivos do teste. Em particular, ela inclui habilidades de comunicação, cooperação e resolução de conflitos (p. ex., lidar de forma construtiva com condições de teste abaixo do ideal ou relatar defeitos aos desenvolvedores).

O desenvolvimento e o teste de software geralmente são feitos por diferentes membros de grupos (diferentes) que coordenam suas tarefas por meio da comunicação. As habilidades de comunicação, a capacidade de trabalhar em equipe e a capacidade de resolver conflitos são necessárias para o sucesso do projeto. No entanto, o nível necessário de habilidades sociais pode ser diferente dependendo do contexto do projeto. Por exemplo, o desenvolvimento ágil de software pode exigir mais habilidades sociais do que os modelos de desenvolvimento sequenciais centrados em documentos, bem como testes externos.

Competência Pessoal

A eficácia e a eficiência dos membros da equipe de teste também dependem de sua capacidade e vontade de se desenvolver, de suas habilidades e de suas atitudes. Por exemplo, trabalhar em uma equipe ágil auto-organizada pode exigir um nível mais alto de autogerenciamento e disciplina de todos os membros da equipe, enquanto um Gerente de Teste de uma equipe de teste hierárquica, por exemplo, precisa ser capaz de delegar trabalho. Muitas vezes, é necessário muita confiabilidade e resiliência, principalmente em projetos de tempo crítico. Além disso, a disposição para ajudar, para aprender e a abertura para mudanças são importantes em um processo de mudança em todos os modelos de SDLC.

3.1.3 Avaliação das habilidades dos membros da equipe de teste

Em muitos casos, as equipes de teste são formadas com o pessoal existente. Para entender as capacidades dos membros da equipe e a necessidade de desenvolvimento pessoal, o gerenciamento de testes precisa avaliar as habilidades da equipe de testes existente e compará-las com as habilidades necessárias, que podem ser documentadas em uma matriz de habilidades.

Existem modelos para ajudar as equipes e os membros da equipe a trabalhar de forma mais eficaz (p. ex., "Team Roles" de Meredith Belbin, DISG® ou PCM®). De acordo com Belbin (Belbin, 2010), as equipes funcionam de forma eficaz quando são formadas por diferentes tipos de personalidade e funções. Esses modelos ajudam as equipes a identificarem as habilidades que possuem e as que podem estar faltando.

A competência profissional e metodológica dos membros da equipe de teste pode ser avaliada por meio da demonstração de tarefas típicas de teste:

- Esboço de uma estratégia de teste e discutir o feedback com os colegas;
- Revisar a base do teste e comunicar as descobertas, o que também pode revelar habilidades de comunicação;
- Determinar as técnicas de teste para atingir objetivos de teste específicos para um determinado contexto de projeto;
- Aplicar adequadamente as várias técnicas de teste
- Redigir um relatório de conclusão do teste relatório que inclua uma avaliação dos resultados do teste

Além disso, as habilidades podem ser avaliadas por meio de credenciais externas, certificações, experiência profissional e diplomas.

Especialmente no desenvolvimento ágil de software, as equipes identificam as habilidades necessárias participando regularmente de retrospectivas e recebendo feedback. Orientadores ou mentores experientes os ajudam a desenvolver suas habilidades e a identificar e resolver lacunas de conhecimento.

3.1.4 Desenvolvimento das habilidades dos membros da equipe de teste

Uma equipe de teste pode não ter todas as habilidades necessárias no início de um projeto. Embora um conjunto perfeito de indivíduos possa não estar disponível, uma equipe forte pode equilibrar os pontos fortes e fracos dos indivíduos.

O gerenciamento de testes ou a equipe de testes pode identificar as necessidades de desenvolvimento necessárias comparando as habilidades exigidas com as disponíveis em uma matriz de habilidades. Com base nisso, eles podem determinar as abordagens para o desenvolvimento de competências:

- O treinamento e a educação ensinam conhecimentos e práticas predefinidos, geralmente em uma sala de aula (virtual) (p. ex., enviando pessoas para um curso de treinamento, realizando sessões de treinamento internamente, desenvolvendo treinamento personalizado ou usando cursos de e-learning ao vivo).
- O autoestudo é uma forma de aprender sobre um assunto que envolve estudar sozinho, em vez de em uma sala de aula (virtual) (p. ex., ler livros, assistir a vídeos gravados ou pesquisar recursos na Internet).
- Aprendizagem entre pares, na qual os colegas compartilham conhecimentos, ideias e experiências e aprendem uns com os outros.
- Mentoring ou coaching são abordagens em que um membro da equipe que é novo em uma função recebe orientação individual de um coach, ou conhecimento, habilidades e/ou experiência de um mentor experiente. A pessoa experiente atua como um recurso contínuo para fornecer conselhos e assistência.
- O treinamento no trabalho também é bem conhecido e é uma mistura de autoaprendizagem, aprendizado entre colegas e orientação ou treinamento.

Nem todas as abordagens para o desenvolvimento de competências são igualmente eficazes e eficientes. O estudo autônomo e o treinamento, por exemplo, são bem adequados para o desenvolvimento de competências profissionais e metodológicas. Por esse motivo, o conhecimento básico em testes pode ser desenvolvido por meio da participação em sessões de treinamento do ISTQB® ou por meio de estudo autônomo com base nos programas de estudos do ISTQB® . Entretanto, para desenvolver a competência social e pessoal, recomenda-se usar abordagens como treinamento e coaching, que geralmente são mais promissoras do que o estudo autônomo. A troca social, o feedback e a reflexão estão entre os principais fatores de sucesso para desenvolver a competência social e pessoal.

3.1.5 Habilidades gerenciais necessárias para gerenciar uma equipe de teste

Qualquer pessoa que queira liderar com sucesso uma equipe de teste deve ter habilidades de gerenciamento. Elas incluem competência profissional e metodológica em atividades fundamentais de gerenciamento (p. ex., planejamento, monitoramento do progresso, controle e relatórios). São necessários conhecimentos e habilidades específicos de gerenciamento de testes para a realização de testes (p. ex., conhecimento de diferentes abordagens de teste, desenvolvimento de estratégias de teste ou uso de técnicas de teste ou do SDLC aplicado).

Liderar ou treinar uma equipe de teste significa agir adequadamente no relacionamento com outros membros da equipe de teste e ter a capacidade e a disposição de se desenvolver em circunstâncias variáveis. Por esse motivo, a competência social e pessoal são fatores essenciais de sucesso para liderar uma equipe de teste. Isso inclui a resiliência, a capacidade de delegar e a capacidade de ser aceito pela equipe de teste. Além disso, inclui a capacidade de afirmar os interesses do teste no projeto, de promover os benefícios do teste e de se comunicar e resolver conflitos com todos os stakeholders.

Para adquirir pessoas para a equipe de teste, são necessárias habilidades de análise das condições sociais, de equipe e de trabalho. Essas habilidades ajudam a garantir que a equipe de teste se adapte às condições de trabalho ou, se possível, que as condições de trabalho sejam adaptadas à equipe de teste. Além disso, as equipes de teste estão sujeitas a processos de desenvolvimento dinâmicos e, portanto, exigem habilidades situacionais. (p. ex., de acordo com as fases do modelo Tuckman de desenvolvimento de pequenos grupos) :

- Disposição para ajudar os membros da equipe de teste a se juntarem à equipe de teste (Forming);
- A capacidade de resolver conflitos dentro da equipe de teste (Storming);
- Disciplina e liderança orientada por metas para garantir valores e regras acordados (Norming);
- A capacidade de delegar para dar responsabilidade pessoal à equipe de teste (Performing);
- A capacidade de agir com apreço e confiança com os membros da equipe de teste que estão saindo (Adjourning);

3.1.6 Fatores motivadores ou desmotivadores para uma equipe de teste em determinadas situações

Membros da equipe de teste satisfeitos e motivados aumentam a produtividade e a performance e, portanto, têm um impacto significativo no sucesso dos projetos. Quando isso é alcançado, o treinamento cruzado ocorre informalmente, os membros da equipe de teste podem gerenciar sua própria carga de trabalho e o gerenciamento de testes tem mais tempo para lidar com questões externas. A Teoria de Dois Fatores (motivation-hygiene theory (Herzberg, et al., 1993)) faz distinção entre motivadores e fatores de higiene:

Os motivadores são percebidos conscientemente e podem levar ao crescimento e à satisfação. Isso pode incluir:

- Reconhecimento e valorização do trabalho realizado (p. ex., incentivos e qualquer outra abordagem individual que os membros da equipe de teste considerem gratificante)
- Maior responsabilidade e autonomia (p. ex., para definir os processos de teste em uma equipe de teste)
- Tarefas interessantes, significativas e desafiadoras que os membros da equipe de teste percebem como possíveis e, ao mesmo tempo, pelas quais vale a pena se esforçar (p. ex., a seleção e a introdução de uma nova ferramenta para automação de testes)
- Avanço e desenvolvimento profissional (p. ex., o desenvolvimento de um testador experiente para um Gerente de Teste ou proprietário de processo de teste)

Os fatores de higiene geralmente são considerados óbvios. O cumprimento deles não leva automaticamente a uma maior satisfação. Se estiverem ausentes, podem ter um efeito desmotivador sobre os membros da equipe de teste:

- Remuneração adequada (p. ex., salário de mercado, horas extras pagas, bons benefícios sociais);
- Política de valorização do pessoal e estilo de gerenciamento (p. ex., gerenciamento enxuto, metas realistas, proteção contra acesso externo e sobrecarga);
- Condições de trabalho agradáveis (p. ex., especificações não ambíguas, objetos de teste maduros, defeitos adequadamente corrigidos, local de trabalho apropriado, ambiente de teste estável);
- Segurança como uma necessidade existencial (p. ex., local de trabalho seguro e cumprimento de acordos);
- Bom relacionamento interpessoal (p. ex., com colegas de trabalho, supervisores).

Consequentemente, o gerenciamento de testes deve eliminar continuamente os fatores de desmotivação e, ao mesmo tempo, criar e fortalecer os fatores de motivação.

Mais informações podem ser encontradas em (Belbin, 2010) (Marston, 1999) (Kahler, 2008).

3.2 Relacionamentos com os stakeholders

Introdução

No gerenciamento de testes, é importante otimizar os testes para oferecer um bom valor comercial. O excesso de testes pode resultar em atrasos e custos excessivos que superam os benefícios, enquanto a insuficiência de testes pode levar à entrega de um produto de baixa qualidade aos usuários. A abordagem ideal está entre esses dois extremos. É responsabilidade do Gerente de Teste ajudar os stakeholders a entender esse equilíbrio e o valor agregado dos testes para alcançar esse equilíbrio, tendo em mente, por exemplo, as restrições de tempo típicas de um projeto.

3.2.1 Custo da qualidade

Os benefícios dos testes são compensados pelos custos de qualidade. Um meio de quantificar o custo total dos esforços e defeitos relacionados à qualidade é chamado de custo da qualidade. O custo da qualidade envolve a

classificação dos custos operacionais e do projeto em quatro categorias relacionadas aos custos de defeitos do produto:

- **Custos de Prevenção de Defeitos:** O custo de todas as atividades planejadas e proativas para evitar a má qualidade (p. ex., qualificação dos desenvolvedores para suas tarefas, como treinamento na criação de código seguro ou de fácil manutenção, revisão da base de testes o mais cedo possível e comunicação adequada dentro da equipe).
- **Custos de Avaliação:** O custo de todas as atividades voltadas para a detecção de defeitos (p. ex., a realização de testes estáticos e dinâmicos e a revisão de produtos de trabalho).
- **Custos de Falhas Internas:** O custo de todas as atividades reativas (p. ex., correção de defeitos encontrados durante o teste, fornecimento de soluções alternativas).
- **Custos de Falhas Externas:** O custo de todas as atividades reativas e sem valor agregado (p. ex., perda de receita, ativos, saúde humana, vida humana ou meio ambiente, custos legais relacionados à correção de defeitos, testes, implantação e suporte devido à entrega de produtos defeituosos ao cliente ("pós-lançamento"), correção de defeitos de campo (sinalizados pelos clientes)).

Os custos totais de avaliação e os custos de falha interna geralmente são significativamente menores do que os custos de falha externa. Portanto, isso torna os testes extremamente valiosos. Ao determinar os custos nessas quatro categorias, os gerentes de teste podem criar um caso comercial convincente para os testes.

Há mais abordagens que podem ser consideradas para definir o custo da qualidade. O syllabus do ISTQB® oferece suporte a duas delas. Esse syllabus é baseado na abordagem de Feigenbaum, e o *ISTQB® Foundation Level Syllabus v4* apresenta a abordagem de Boehm (consulte o *ISTQB® Foundation Level Syllabus v4*, seção 1.3, *Princípios de Teste*). Essas duas abordagens foram selecionadas para alcançar uma compreensão mais ampla do custo da qualidade. A abordagem de Feigenbaum (Feigenbaum, Nov/Dec 1956) considera a qualidade como um processo orientado para o cliente e para toda a empresa, enquanto a abordagem de Boehm (Boehm, 1979) se concentra na compensação entre o custo da prevenção e o custo da falha no desenvolvimento de software (Hadjicostas, 2004).

3.2.2 Relação custo-benefício dos testes

Embora a maioria das organizações considere os testes valiosos em algum sentido, poucos gerentes, incluindo os gerentes de teste, conseguem quantificar, descrever ou articular esse valor. Além disso, muitos gerentes de teste, líderes de teste e testadores se concentram nos detalhes operacionais dos testes (ou seja, aspectos específicos da tarefa de teste ou do nível de teste), ignorando as questões táticas e estratégicas mais amplas (de nível mais alto) relacionadas aos testes, com as quais outros stakeholders, especialmente os gerentes, se preocupam.

Os testes trazem benefícios para a organização, o projeto e/ou a operação de forma quantitativa e qualitativa:

- **Benefícios Qualitativos:** incluem melhor reputação de qualidade, liberações mais suaves e previsíveis, maior confiança, proteção contra responsabilidade legal e redução do risco de perda de missões inteiras ou até mesmo de vidas.
- **Benefícios Quantitativos:** incluem defeitos encontrados, evitados ou corrigidos antes do lançamento, defeitos conhecidos antes do lançamento (ou seja, não corrigidos, mas documentados, talvez com soluções alternativas), benefícios de custo (Bohm 1981, Böhler 2008), redução do nível de risco com a execução de testes e fornecimento de informações sobre o status do projeto, do processo e do produto.

Um benefício adicional dos testes é que todas os stakeholders obtêm informações adequadas para tomar decisões informadas sobre se a qualidade do produto é suficiente para entrar em operação, com ou sem defeitos. Às vezes, entrar em operação com defeitos conhecidos é muito melhor do que esperar até que os defeitos sejam resolvidos. Nesses casos em que um defeito pode ser tolerado, isso depende muito da probabilidade de ocorrência e da gravidade do defeito.

O custo da qualidade por defeito de teste é calculado da seguinte forma[3]:

Economia Média por Defeito (EMD)

$$EMD = Média\ de\ CFex - (Média\ de\ CAv + Média\ de\ CFin)$$

Onde:

- **CFex**: Custo de Falhas Externas
- **CAv**: Custos de Avaliação
- **CFin**: Custo de Falhas Internas

Custo Total da Qualidade (CTQ)

$$CTQ = CPD + (CAv \times NDalb) + (CFin \times NDaln) + (CFex \times NDpln)$$

Onde:

- **CPD**: Custo de Prevenção de Defeitos
- **CAv**: Custo de Avaliação
- **NDalb**: Número de Defeitos encontrados antes da liberação
- **CFin**: Custo de Falhas internas
- **NDaln**: Número de Defeitos encontrados antes do lançamento
- **CFex**: Custo de Falhas externas
- **NDpln**: Número de Defeitos encontrados após o lançamento

Como exemplo, suponha que você tenha calculado o seguinte custo de qualidade por defeito para um produto:

- Custo de prevenção de defeitos (CPD): $180
- Custo médio de avaliação por defeito (CAv): $ 500
- Média do custo de falhas internas por defeito (CFin): $ 200
- Média do custo de falhas externas por defeito (CFex): $ 4.000

Os custos médios de prevenção de defeitos, custos de avaliação e custos de falha interna são calculados usando o número de defeitos encontrados antes do lançamento, enquanto os custos médios de falha externa são calculados usando o número de defeitos encontrados após o lançamento. Com esses valores, podemos calcular a economia média por defeito da seguinte forma:

Economia média por defeito (EMD)

$$EMD = \$\,4.000 - (\$\,500 + \$\,200) = \$\,\mathbf{3.300}$$

[3] ***Nota do BSTQB**: a apresentação das fórmulas foi adequada para facilitar o entendimento do leitor. Caso tenha interesse, consulte a versão em inglês deste syllabus no site do ISTQB.*

A curva de Boehm é uma representação gráfica dos custos de correção de defeitos ao longo do tempo no SDLC. Isso conclui que os testes devem ser feitos mais cedo no SDLC para reduzir o custo de correção de defeitos. A curva de Boehm mostra que os custos de falhas internas ou o custo de correção de um defeito, aumenta quanto mais tarde um defeito é descoberto no SDLC. Usando essas informações, o Gerente de Teste deve se esforçar para encontrar a relação ideal entre os custos de prevenção de defeitos e os custos internos e externos. versus os custos internos e externos.

O esforço de teste deve se basear no risco específico do projeto e do produto e no risco que a empresa está disposta a correr. O excesso de testes pode gerar custos mais altos do que o benefício da redução do nível de risco. redução do nível de risco. Se for testado muito pouco, os defeitos não detectados podem representar um alto risco e gerar custos mais altos do que os custos dos testes omitidos. Os testes baseados em riscos (consulte a seção 1.3, *Teste Baseado em Risco*) apoiam a relação custo-benefício dos testes, investindo níveis de esforço de teste proporcionais aos níveis de risco e priorizando os testes com base em seus níveis de risco.

Os gerentes de teste devem entender quais desses benefícios e custos se aplicam à sua organização, projeto e/ou operação e ser capazes de comunicar o valor agregado dos testes em termos desses benefícios e do custo da qualidade por defeito.

4 Referências

Normas e Padrões

IEC 61508 (2010) Functional safety of electrical/electronic/programmable electronic safety-related systems - Partes 1 a 7

ISO/IEC/IEEE 29119-2 (2021) ISO/IEC/IEEE 29119-2 Software and systems Engineering-Software testing-Parte 2 Test processes

ISO/IEC/IEEE 29119-3 (2021) ISO/IEC/IEEE 29119-3 Software and systems Engineering-Software testing-Parte 3 Test documentation

Documentos ISTQB®

ISTQB® Certified Tester Ágil Test Leadership at Scale Syllabus v2.0 (2023)

ISTQB® Certified Tester Foundation Level Syllabus V.4 (2023)

ISTQB® Certified Tester Expert Level Test Management Syllabus v1.0 (2011)

ISTQB® Certified Tester Expert Level Improving the Test Process Syllabus v1.0.1 (2011)

Livros

Basili, V., Trendowicz, A., Kowalczyk, M., Heidrich, J., Seaman, C., Münch, J., & Rombach, D. (2014). Aligning Organizations Through Measurement – The GQM+ Strategies Approach. Springer International.

Bath, G., & van Veenendaal, E. (2014). Improving the Test Process - chapter 6: Process for Improvement. Rocky Nook.

Belbin, R. M. (2010). Management Teams: Why They Succeed or Fail. London: Routledge.

Black, R. (2009). Managing the Testing Process, 3rd Edition. John Wiley & Sons.

Boehm, B. (1979). Software Engineering Economics. Prentice-Hall.

Bonebright, D. A. (2010). 40 years of storming: a historical review of Tuckman's model of small group development (1 Ausg., Bd. 13). Human Resource Development International, 1, 2010, Vol. 13.

Craig, R., & Jaskiel, S. P. (2002). Systematic Software Testing. Artech House.

Derby, E., & Larsen, D. (2006). Agile Retrospectives – Making Good Teams Great,. The Pragmatic Bookshelf.

Erpenbeck, J., & von Rosenstiel, L. (2017). Handbuch Kompetenzmessung. Stuttgart: Schäffer-Poeschel.

Fowler, M. (2010). Hybrid development processes. IEEE Software, 27(2), 57-63.

Herzberg, F., Mausner, B., & Bloch Snyderman, B. (1993). Motivation to Work. London: Routledge.

Kahler, T. (2008). The Process Therapy Model: The Six Personality Types with Adaptations. Taibi Kahler Associates, Inc.

Marston, W. M. (1999). Emotions Of Normal People. London: Routledge.

Sonntag, K., & Schmidt-Rathjens, C. (2005). Anforderungsanalyse und Kompetenzmodelle. Wiesbaden: VS Verlag für Sozialwissenschaften.

Tuckman, B. W. (1965). Developmental sequence in small groups (Bd. 63(6)). Psychological Bulletin.

van Ewijk, A. (2013). TPI NEXT – Business Driven Test Process Improvement, Sogeti Nederland B.V.

van Solingen, R., & Berghout, E. (1999). The Goal Question Metric Method – A Practical Guide for Quality Improvement of Software Development. McGraw-Hill.

van Veenendaal, E. (2012). The PRISMA Approach: Practical Risk-Based Testing. UTN Publishers.

van Veenendaal, E. (2020). TMMi in the Agile world, version 1.4. TMMi Foundation.

van Veenendaal, E., & Cannegieter, J. J. (2011). The Little TMMi – Objective-Driven Test Process Improvement. UTN Publishers.

Artigos

Feigenbaum, Armand V. (Nov/Dec 1956) Harvard Business Review, Vol. 34 Issue 6, p93-101

Hadjicostas, Evsevios (2004) Total Quality Management and Cost of Quality, Springer https://link.springer.com/chapter/10.1007/978-3-662-09621-5_7

Websites e Web Pages

www.tmmi.org Test Maturity Model integration (TMMi®); last visit January 31st, 2024

www.tmap.net Test Process Improvement (TPI); last visit January 31st, 2024

www.wikipedia.org/wiki/PDCA Plan-Do-Check-Act; last visit January 31st, 2024

https://glossary.istqb.org Glossário de Termos de Teste do ISTQB

https://bstqb.org.br Site da Conselho Membro do ISTQB no Brasil

As referências anteriores apontam para informações disponíveis na Internet e em outros lugares. Embora essas referências tenham sido verificadas no momento da publicação deste syllabus, o ISTQB® não pode ser responsabilizado caso as referências não estejam mais disponíveis.

5 Apêndice A – Objetivos de Aprendizagem e Níveis Cognitivos

Os objetivos específicos de aprendizado que se aplicam a este syllabus são mostrados no início de cada capítulo. Cada tópico do syllabus será examinado de acordo com seu objetivo de aprendizado.

Os objetivos de aprendizado começam com um verbo de ação correspondente ao seu nível cognitivo de conhecimento, conforme listado abaixo.

Nível 1: Lembrar (K1)

O candidato se lembrará, reconhecerá e recordará um termo ou conceito.

Verbos de ação: Recordar, reconhecer.

Exemplos
Relembrar os conceitos da pirâmide de teste.
Reconhecer os objetivos típicos dos testes.

Nível 2: Compreender (K2)

O candidato pode selecionar as razões ou explicações para declarações relacionadas ao tópico e pode resumir, comparar, classificar e dar exemplos para o conceito de teste.

Verbos de ação: Classificar, comparar, diferenciar, distinguir, explicar, dar exemplos, interpretar, resumir

Exemplos	Notas
Classificar as ferramentas de teste de acordo com sua finalidade e as atividades de teste que elas suportam.	
Comparar os diferentes níveis de teste.	Pode ser usado para procurar semelhanças, diferenças ou ambos.
Diferenciar teste de depuração.	Procura diferenças entre os conceitos.
Distinguir entre riscos do projeto e do produto.	Permite que dois (ou mais) conceitos sejam classificados separadamente.
Explicar o impacto do contexto no processo de teste.	
Dar exemplos de por que os testes são necessários.	
Inferir a causa raiz dos defeitos a partir de um determinado perfil de falhas.	
Resumir as atividades do processo de revisão do produto de trabalho.	

Nível 3: Aplicar (K3)

O candidato pode executar um procedimento quando confrontado com uma tarefa familiar ou selecionar o procedimento correto e aplicá-lo a um determinado contexto.

Verbos de ação: Aplicar, implementar, preparar, usar

Exemplos	Notas

Aplicar a análise de valor limite para derivar casos de teste a partir de determinados requisitos.	Deve se referir a um procedimento/técnica/processo etc.
Implementar métodos de coleta de métricas para dar suporte aos requisitos técnicos e gerenciais.	
Preparar testes de instalação para aplicativos móveis.	
Usar a rastreabilidade para monitorar o progresso do teste para garantir a integridade e a consistência com os objetivos do teste, a estratégia do teste e o plano de teste.	Pode ser usado em um LO que deseja que o candidato seja capaz de usar uma técnica ou procedimento. Semelhante a "aplicar".

Nível 4: Analisar (K4)

O candidato pode separar as informações relacionadas a um procedimento ou técnica em suas partes constituintes para melhor compreensão e pode distinguir entre fatos e inferências. Uma aplicação típica é analisar um documento, software ou situação de projeto e propor ações apropriadas para resolver um problema ou concluir uma tarefa.

Verbos de ação: Analisar, desconstruir, delinear, priorizar, selecionar

Exemplos	Notas
Analisar uma determinada situação de projeto para determinar quais técnicas de teste de caixa preta ou baseadas em experiência devem ser aplicadas para atingir objetivos específicos.	Examinável somente em combinação com um objetivo mensurável da análise. Deve ser do tipo "Analisar xxxx para xxxx" (ou similar).
Priorizar os casos de teste em um determinado conjunto de testes para execução com base nos riscos relacionados ao produto.	
Selecionar os níveis de teste e os tipos de teste adequados para verificar um determinado conjunto de requisitos.	Necessário quando a seleção requer análise.

Referência

(Para os níveis cognitivos dos objetivos de aprendizado)

Anderson, L. W. e Krathwohl, D. R. (eds) (2001) A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives (Uma revisão da taxonomia de Bloom dos objetivos educacionais), Allyn & Bacon

Os objetivos específicos de aprendizado que se aplicam a este syllabus são mostrados no início de cada capítulo.

6 Apêndice B - Matriz de rastreabilidade de resultados *de negócio* com objetivos de aprendizagem

Esta seção lista a rastreabilidade entre os resultados comerciais e os objetivos de aprendizagem do programa de gerenciamento de testes de nível avançado.

Resultados de negócio: Gerenciamento de testes de nível avançado		**BO1**	**BO2**	**BO3**	**BO4**	**BO5**	**BO6**	**BO7**	**BO8**	**BO9**	**BO10**	**BO11**
TM_01	Gerenciar testes em vários projetos de desenvolvimento de software aplicando processos de gerenciamento de testes estabelecidos para a equipe do projeto ou organização de testes	12										
TM_02	Identificar os stakeholders no teste e os modelos de ciclo de vida de desenvolvimento de software que são relevantes em um determinado contexto		4									
TM_03	Organizar a identificação de riscos e avaliação de riscos em qualquer ciclo de vida de desenvolvimento de software e usar esses resultados para orientar os testes a fim de atingir os objetivos do teste			6								
TM_04	Definir uma estratégia de teste do projeto consistente com a estratégia de teste organizacional e o contexto do projeto				11							
TM_05	Monitorar e controlar continuamente os testes para atingir os objetivos do projeto					4						
TM_06	Avaliar e relatar o progresso do teste aos stakeholders do projeto						3					
TM_07	Identificar as habilidades necessárias e desenvolvê-las em sua equipe							6				
TM_08	Preparar e apresentar um caso de negócio para testes em diferentes contextos que descreva os custos e os benefícios esperados								5			

Resultados de negócio: Gerenciamento de testes de nível avançado		BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8	BO9	BO10	BO11
TM_09	Liderar as atividades de aprimoramento do processo de teste e as atividades de melhoria do processo de teste em projetos ou fluxos de produtos de desenvolvimento de software, e contribuir para iniciativas organizacionais de melhoria do processo de teste									5		
TM_10	Planejar as atividades de teste e estimar o esforço de teste										9	
TM_11	Criar relatórios de defeitos e um fluxo de trabalho de defeitos adequado para um ciclo de vida de desenvolvimento de software											6

Unique LO	Objetivo de aprendizado	K-Level	BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8	BO9	BO10	BO11
1	Gerenciamento das atividades de teste												
1.1	O Processo de Teste												
TM-1.1.1	Resumir o planejamento de testes	K2	X			X							
TM-1.1.2	Resumir o monitoramento de testes e controle de testes	K2	X				X						
TM-1.1.3	Resumir a conclusão do teste	K2	X					X					
1.2	O Contexto do Teste												
TM-1.2.1	Comparar por que diferentes stakeholders estão interessados em testes	K2		X		X							
TM-1.2.2	Explicar por que o conhecimento dos stakeholders é importante no gerenciamento de testes	K2		X		X							
TM-1.2.3	Explicar os testes em um modelo de desenvolvimento de software híbrido	K2		X		X							
TM-1.2.4	Resumir as atividades de gerenciamento de testes para vários ciclos de vida de desenvolvimento de software	K2	X	X		X							
TM-1.2.5	Comparar as atividades de gerenciamento de testes em vários níveis de teste	K2	X			X							
TM-1.2.6	Comparar as atividades de gerenciamento de testes para vários tipos de testes	K2	X			X							
TM-1.2.7	Analisar um determinado projeto e determinar as atividades de gerenciamento de testes, enfatizando o planejamento de testes, monitoramento de testes, e controle de testes	K4	X			X							
1.3	Teste Baseado em Risco												

Unique LO	Objetivo de aprendizado	K-Level	BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8	BO9	BO10	BO11
TM-1.3.1	Explicar as várias medidas que os testes baseados em riscos são tomadas para responder aos riscos	K2			X								
TM-1.3.2	Dar exemplos de diferentes técnicas que um Gerente de Teste pode usar para identificar riscos relacionados à qualidade do produto	K2			X								
TM-1.3.3	Fazer um resumo dos fatores que determinam os níveis de risco relacionados à qualidade do produto	K2			X								
TM-1.3.4	Selecionar as atividades de teste adequadas para mitigar os riscos de acordo com seu nível de risco em um determinado contexto	K4			X								
TM-1.3.5	Diferenciar entre exemplos pesados e leves de testes baseados em riscos	K2			X								
TM-1.3.6	Dar exemplos de métricas de sucesso e dificuldades associadas aos testes baseados em riscos	K2			X								
1.4	A Estratégia de Teste do Projeto												
TM-1.4.1	Explicar as opções típicas de uma abordagem de teste	K2				X							
TM-1.4.2	Analisar uma estratégia de teste organizacional e o contexto do projeto para selecionar a abordagem de teste apropriada	K4				X							
TM-1.4.3	Usar a metodologia de metas S.M.A.R.T. para definir objetivos de teste mensuráveis e critérios de saída	K3				X							
1.5	Melhoria do Processo de Teste												
TM-1.5.1	Explicar como usar o modelo IDEAL para aprimorar o processo de teste em um determinado projeto	K2									X		

Unique LO	Objetivo de aprendizado	K-Level	BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8	BO9	BO10	BO11
TM-1.5.2	Resumir a abordagem de melhoria baseada em modelos para testar a melhoria do processo e entender como aplicá-la em um contexto de projeto	K2									X		
TM-1.5.3	Resumir a abordagem de melhoria baseada em análise para testar a melhoria do processo e entender como aplicá-la em um contexto de projeto	K2									X		
TM-1.5.4	Implementar uma retrospectiva de projeto ou iteração para avaliar os processos de teste e descobrir áreas de teste a serem aprimoradas	K3									X		
1.6	Ferramentas de Teste												
TM-1.6.1	Resumir as práticas recomendadas para a introdução de ferramentas	K2										X	
TM-1.6.2	Explicar o impacto de diferentes aspectos técnicos e comerciais ao decidir sobre um tipo de ferramenta	K2										X	
TM-1.6.3	Analisar uma determinada situação para criar um plano de seleção de ferramentas, abrangendo riscos, custos e benefícios	K4										X	
TM-1.6.4	Diferenciar os estágios do ciclo de vida da ferramenta	K2										X	
TM-1.6.5	Dar exemplos de métricas coleta e avaliação por meio de ferramentas	K2									X	X	
2	Gerenciando o Produto												
2.1	Métricas de Teste												
TM-2.1.1	Dar exemplos de métricas para atingir os objetivos do teste	K2					X						

Unique LO	Objetivo de aprendizado	K-Level	BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8	BO9	BO10	BO11
TM-2.1.2	Explicar como controlar o progresso do teste usando métricas de teste	K2					X						
TM-2.1.3	Analisar os resultados dos testes para criar relatórios de testes que capacitem os stakeholders a tomar decisões	K4					X	X					
2.2	Estimativa de Teste												
TM-2.2.1	Explicar os fatores que precisam ser considerados na estimativa de testes	K2	X							X		X	
TM-2.2.2	Dar exemplos de fatores que podem influenciar as estimativas de teste	K2	X							X		X	
TM-2.2.3	Selecionar uma técnica ou abordagem apropriada para a estimativa de testes em um determinado contexto	K4	X							X		X	
2.3	Gerenciamento de Defeitos												
TM-2.3.1	Implementar um processo de gerenciamento de defeitos, incluindo o fluxo de trabalho de defeitos que pode ser usado para monitorar e controlar defeitos	K3											X
TM-2.3.2	Explicar o processo e os participantes necessários para o gerenciamento eficaz de defeitos	K2											X
TM-2.3.3	Explicar as especificidades do gerenciamento de defeitos no desenvolvimento de software Ágil	K2	X										X
TM-2.3.4	Explicar os desafios do gerenciamento de defeitos no desenvolvimento de software híbrido	K2	X										X
TM-2.3.5	Usar os dados e as informações de classificação que devem ser coletados durante o gerenciamento de defeitos	K3											X

Unique LO	Objetivo de aprendizado	K-Level	BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8	BO9	BO10	BO11
TM-2.3.6	Explicar como as estatísticas de relatórios de defeitos podem ser usadas para planejar a melhoria do processo	K2										X	X
3	Gerenciando a Equipe												
3.1	A Equipe de Teste												
TM-3.1.1	Dar exemplos de habilidades típicas, necessárias para os membros da equipe de teste em quatro áreas de competência	K2							X				
TM-3.1.2	Analisar um determinado contexto de projeto para determinar as habilidades necessárias para os membros da equipe de teste	K4							X				
TM-3.1.3	Explicar as técnicas típicas de avaliação de habilidades dos membros da equipe de teste	K2							X				
TM-3.1.4	Diferenciar as abordagens típicas para desenvolver as habilidades dos membros da equipe de teste	K2							X				
TM-3.1.5	Explicar as habilidades necessárias para gerenciar uma equipe de teste	K2							X				
TM-3.1.6	Dar exemplos de fatores de motivação e higiene para os membros da equipe de teste	K2							X				
3.2	Relacionamentos com Stakeholders												
TM-3.2.1	Dar exemplos de cada uma das quatro categorias que determinam o custo da qualidade	K2								X			
TM-3.2.2	Aplicar um cálculo de custo-benefício para estimar o valor agregado dos testes para os stakeholders	K3						X		X			

7 Apêndice C - Notas de versão

O *ISTQB® Advanced Level Test Management Syllabus v3.0* é uma grande atualização baseada no *Advanced Level Syllabus Test Manager 2012*. Por esse motivo, não há notas de versão detalhadas por capítulo e seção. No entanto, um resumo das principais alterações é fornecido abaixo.

Nesta versão, todos os Objetivos de Aprendizagem (OAs) foram editados para torná-los atômicos e para criar rastreabilidade um a um entre os OAs e as seções do syllabus, de modo que não haja conteúdo sem que haja também um OA. A meta é tornar esta versão mais fácil de ler, entender, aprender e traduzir, concentrando-se em aumentar a utilidade prática e o equilíbrio entre conhecimento e habilidades.

Essa versão principal fez as seguintes alterações:

- Redução do tamanho do geral do syllabus. Um syllabus não é um livro-texto, mas um documento que serve para delinear os elementos básicos de um curso avançado sobre teste de software, incluindo os tópicos que devem ser abordados e em que nível. Portanto, em particular:
  - Na maioria dos casos, os exemplos são excluídos do texto. É tarefa do provedor de treinamento fornecer os exemplos, bem como os exercícios, durante o treinamento
  - Foi seguido o "*Syllabus writing checklist*", que sugere o tamanho máximo do texto para os LOs em cada nível K (K2 = máximo de 1.500 caracteres não vazios, K3 = máximo de 2.500 caracteres não vazios, K4 = máximo de 3.000 caracteres não vazios, +/- 20%)
- Redução do número de LOs em comparação com o syllabus CTAL-TM 2012
  - K2: de 39 para 36 LOs
  - K3: de 12 para 5 LOs
  - K4: de 10 para 7 LOs
- A estrutura completa do syllabus foi revisada
- Completo alinhamento com o *ISTQB® Foundation Level Syllabus v4*
- Principais alterações no antigo Capítulo 1 (Processo de Teste) de 2012
  - Restrito ao gerenciamento das atividades de teste (planejamento de teste, monitoramento de teste, controle de teste e conclusão de teste)
  - Integrado como uma seção no novo capítulo "*Gerenciando as Atividades de Teste*"
- Novo capítulo ***Gerenciando as Atividades de Teste***
  - Seção 1.1 - O Processo de Teste: veja acima
  - Seção 1.2 - O Contexto do Teste: ampliada para abranger o desenvolvimento de software não sequencial modelos
  - Seção 1.3 - Teste Baseado em Risco: completamente reescrita para torná-la mais aplicável em nível de projeto
  - Seção 1.4 - A Estratégia de Teste do Projeto: como o plano de teste já está definido no *ISTQB® Foundation Level Syllabus v4*, o foco está na seleção da abordagem de teste adequada e em como definir objetivos de teste mensuráveis

- Seção 1.5 – Melhoria do Processo de Teste: integrando-o ao gerenciamento das atividades de teste, mostrando como aplicá-lo em um contexto de projeto e implementando-o usando retrospectivas em uma iteração ou projeto
- Seção 1.6 - Ferramentas de Teste: a introdução de ferramentas foi movida do *ISTQB® Foundation Level Syllabus v3.1* (não está presente no *ISTQB® Foundation Level Syllabus v4*)

- Novo capítulo ***Gerenciando o Produto***
  - Seção 2.1 - Métricas de Teste: seções anteriores que definem métricas e uso de métricas
  - Seção 2.2 - Estimativa de Teste: O *ISTQB® Foundation Level Syllabus v4* já cobre o cálculo da estimativa de teste. Expandido para selecionar, em um nível K4, técnicas adequadas de estimativa de teste e o uso de estimativas de teste nos modelos SDLC
  - Seção 2.3 - Gerenciamento de Defeitos: alinhado com as edições mais recentes dos padrões e expandido para uso no desenvolvimento de software Ágil e híbrido
- Novo capítulo ***Gerenciando a Equipe***
  - Seção 3.1 - A Equipe de Teste: os principais tópicos são os mesmos do syllabus TM 2012. Identifique as habilidades individuais e componha as equipes de teste
  - Seção 3.2 - Relacionamentos com Stakeholders: é a antiga seção 2.7 do syllabus CTAL-TM 2012 intitulada "Valor de Negócio dos Testes"
- Principais alterações e seções/capítulos excluídos no CTAL-TM Syllabus 2012
  - Removida a seção sobre testes distribuídos, terceirizados e internos
  - Removida a seção sobre gerenciamento da aplicação de padrões do setor
  - Removido o capítulo sobre revisões
  - Removidas as subseções sobre aprimoramento do processo de teste CTP e STEP
  - Removidas as subseções sobre análise de teste, projeto de teste, implementação de teste e execução de teste

8 Apêndice D - Palavras-chave específicas do domínio

Nome do termo	Definição
Goal Question Metric (GQM)	Uma abordagem para a medição de software usando um modelo de três níveis que consiste em um nível conceitual (meta), um nível operacional (pergunta) e um nível quantitativo (métrica).
IDEAL	Um modelo de melhoria organizacional que serve como um roteiro para iniciar, planejar e implementar ações de melhoria.
indicador	Uma medida que fornece uma estimativa ou avaliação de atributos especificados derivados de um modelo com relação às necessidades de informações definidas.
medida	O número ou a categoria atribuída a um atributo de uma entidade por meio de uma medição.
métrica	Uma escala de medição e o método usado para medição.
planning poker	Uma técnica de estimativa baseada em consenso, usada principalmente para estimar o esforço ou o tamanho relativo das histórias de usuários no desenvolvimento de software Ágil. É uma variação do método Wideband Delphi que usa um baralho de cartas com valores que representam as unidades nas quais a equipe estima.
Estimativa de três pontos	Uma técnica baseada em especialistas, três estimativas são feitas pelos especialistas: a estimativa mais otimista (o), a estimativa mais provável (m) e a estimativa mais pessimista (p). A estimativa final (E) é a média aritmética ponderada.
Wideband Delphi	Uma técnica de estimativa de teste baseada em especialistas que visa a fazer uma estimativa precisa usando a sabedoria coletiva dos membros da equipe.

9 Apêndice E – Marcas Registradas

CMMI® é marca registrada na U.S. Patent and Trademark Office by Carnegie Mellon University.

ISTQB® é marca registrada de International Software Testing Qualifications Board.

BSTQB® é marca registrada de Brazilian Software Testing Qualifications Board.

TMMi® é marca registrada de TMMi Foundation.

TPI-Next® é marca registrada de Sogeti, The Netherlands.