I•B•Q•T•S

Glossário de Termos utilizados Mundialmente em Testes de Software

Versão 1.1
12 de junho de 2007

Editores: **Martin Tornquist**
Paulo Henrique Nannini

Contribuições

Alan José Nascimento
Carlos Alberto Rodrigues
Daniel José Pereira Queiroz
Daniele Constant Guimarães
Felipe de Oliveira Ângelo
Gabriel Gaudêncio Molina
Martin Tornquist
Paulo Henrique Nannini
Ronaldo de Almeida
Wladimir Lotufo Filho

Este glossário é baseado nos trabalhos de compilação em língua inglesa de muitas pessoas e organizações nos últimos 30 anos destacando-se autores como Myers, Beizer, Kamen, entre outros (vide bibliografia recomendada), de organizações normativas (BS, IEEE, DoD, ISO, entre outras) bem como de organizações de certificação profissional (IBQTS, QAI, ASQ, ALATS, ISTQB, entre outras).
Apoiado pelo "Glossary Working Party" *International Software Testing Qualification Board.*

Conteúdo

Referências Normativas

BS 7925-2:1998. Software Component Testing.

DO-178B:1992. Software Considerations in Airborne Systems and Equipment Certification, Requirements and Technical Concepts for Aviation (RTCA SC167).

IEEE 610.12:1990. Standard Glossary of Software Engineering Terminology.

IEEE 829:1998. Standard for Software Test Documentation.

IEEE 1008:1993. Standard for Software Unit Testing.

IEEE 1012:1986. Standard for Verification and Validation Plans

IEEE 1028:1997. Standard for Software Reviews and Audits.

IEEE 1044:1993. Standard Classification for Software Anomalies.

IEEE 1219:1998. Software Maintenance.

ISO/IEC 2382-1:1993. Data processing - Vocabulary - Part 1: Fundamental terms.

ISO 9000:2000. Quality Management Systems – Fundamentals and Vocabulary.

ISO/IEC 9126-1:2001. Software Engineering – Software Product Quality – Part 1: Quality characteristis and sub-characteristics.

ISO/IEC 12207:1995. Information Technology – Software Life Cycle Processes.

ISO/IEC 14598-1:1996. Information Technology – Software Product Evaluation - Part 1: General Overview.

Definições

A

Aceite: Vide *teste de aceite*.

Adaptabilidade: A capacidade que o sistema[1] tem de ser adaptado para diferentes ambientes especificados sem utilizar ações ou meios que não sejam aqueles fornecidos para esta finalidade. [ISO 9126]. Veja também *portabilidade*.

Adequação: A capacidade que o sistema tem de fornecer um conjunto apropriado de funções para as tarefas e objetivos do usuário especificados. [ISO 9126]. Veja também *funcionalidade.*

Alterabilidade: A capacidade que o sistema tem de permitir que as alterações especificadas sejam implementadas. [ISO 9126]. Veja também *manutenibilidade*.

Alvo do teste: Conjunto de critérios de saída.

Ambiente de teste: Um ambiente que contém hardware, instrumentação, simuladores, ferramentas de software e outros elementos de apoio necessários para conduzir os testes. [Após IEEE 610].

Ambiente operacional: Produtos de hardware e software instalados localmente nos usuários ou clientes onde o sistema em teste será usado. Os softwares podem incluir sistemas operacionais, sistemas de gerenciamento banco de dados e outras aplicações.

Analisabilidade: A capacidade que o sistema tem de ser diagnosticado para encontrar deficiências ou causas das falhas, ou para identificar quais partes que serão modificadas. [ISO 9126]. Veja também *manutenibilidade*.

Analisador: Vide a*nalisador estático*.

Analisador de código: Vide *analisador estático de código*.

[1] O termo "Sistema" está sendo utilizado de forma genérica neste glossário. Ele representa um sistema completo, um software, um produto de software, um programa.

Analisador estático: Ferramenta que realiza a análise estática.

Analisador estático de código: Ferramenta que realiza a análise estática do código. Ela verifica o código fonte em busca de determinadas propriedades, como conformidade a padrões de codificação, métricas de qualidade ou anomalias no fluxo de dados.

Análise de árvore de falhas: Método usado para analisar as causas das falhas (defeitos).

Análise de causa-efeito: Vide *grafo causa-efeito*.

Análise de cobertura: Mensuração da cobertura atingida para um determinado item de cobertura durante a execução dos testes, consultando critérios predeterminados para verificar se são necessários testes adicionais, e, se forem, quais casos de teste são necessários.

Análise de fluxo de dados: Uma forma de análise estática baseada na definição e uso de variáveis.

Análise de impacto: Avaliação das alterações a serem feitas nos documentos de desenvolvimento e teste, e nos sistemas, para implementar essas alterações nos requisitos especificados.

Análise de mudanças: Método que determina a completude da suíte de testes medindo o grau com que essa suíte consegue distinguir entre o programa original e o programa com ligeiras variações (mutações).

Análise de pontos de função (FPA): (Do inglês: *Function Point Analysis*). Método que tem como objetivo medir o tamanho da funcionalidade do sistema. Essa métrica é independente da tecnologia usada e pode ser usada como base para métricas de produtividade, para estimativa de recursos necessários e para controle do projeto.

Análise de pontos de teste (TPA): (do inglês: *Test Point Analysis*) Método de estimativa de esforço de teste baseado nas atividades relacionadas aos testes. Inspirado na análise por pontos de função. [Tmap].

Análise de riscos: Processo de avaliar os riscos identificados para estimar seu impacto e probabilidade de ocorrência.

Análise de valor limite: Técnica de projeto de teste caixa-preta onde os casos de teste são definidos com base no valor limite.

Análise dinâmica: Processo de avaliar o comportamento do sistema durante sua execução, como por exemplo, o desempenho da memória e uso da CPU. [Após IEEE 610].

Análise estática: Análise dos artefatos de software, como por exemplo, requisitos ou código, sem executá-los.

Análise estática do código: Análise do código fonte sem executá-lo.

Anomalia: Qualquer condição que desvia da expectativa (baseada nas especificações de requisitos, documentos de projeto, documentos de usuários, padrões, etc.) ou da percepção ou experiência de alguém. As anomalias podem ser encontradas durante (mas não limitada a) a revisão, testes, análise, compilação ou uso dos sistemas ou da documentação aplicável. [IEEE 1044] Veja também *defeito, desvio, erro, falha, falta, incidente, problema.*

Armazenamento: Vide *utilização de recursos*.

Atratividade: A capacidade que o sistema tem de ser atraente para o usuário. [ISO 9126] Veja também *usabilidade*.

Atributo de qualidade: Um atributo ou característica que afeta a qualidade de um item. [IEEE 610]

Auditoria: Uma avaliação independente do sistema ou processo de software para determinar a conformidade a padrões, diretrizes, especificações, e/ou procedimentos baseados em critérios objetivos, incluindo os documentos que especificam:
(1) A forma ou o conteúdo dos produtos que serão produzidos;
(2) O processo pelo qual os produtos devem ser produzidos;
(3) Como a conformidade a padrões ou diretrizes deve ser medida. [IEEE 1028]

Auditoria de configuração: Verifica o conteúdo das bibliotecas dos itens de configuração, para verificar, por exemplo, a conformidade a padrões. [IEEE 610]

Automação da execução dos testes: Uso de software, como por exemplo, ferramentas de captura/reprodução, para controlar a execução dos testes, a comparação dos resultados obtidos e esperados, a configuração das pré-condições dos testes e outros controles dos testes e funções reportadas.

Automação dos testes: Uso do software para executar ou apoiar as atividades de testes, como por exemplo, gerenciamento, projeto e execução de testes, e verificação dos resultados.

Avaliação: Vide *teste*.

Avaliação heurística: Técnica estática de teste de usabilidade para determinar a conformidade da interface do usuário com os princípios de usabilidade reconhecidos (também chamado de *heurísticas*).

B

Base de teste: Todos os documentos a partir dos quais os requisitos de um sistema podem ser inferidos; documentação na qual os casos de teste são baseados. Se um documento pode ser alterado somente pelo processo formal de alterações, então a base de teste é chamada base de teste congelada. [Após Tmap].

Base de teste congelada: Documento da base de teste que só pode ser alterado por um processo formal de controle de alterações. Veja também *baseline*

***Baseline*:** Uma especificação ou sistema que foi revisado formalmente, ou acordado, e que serve como base para o desenvolvimento posterior. Pode ser mudado somente por um processo formal de controle de alteração. [Após IEEE 610]

Bebugging: Vide *semear erros*. [Abbott]

Bloco básico: Seqüência de um ou mais comandos executáveis consecutivos que não contém ramos.

Boa prática: Um método superior ou uma prática inovadora que contribui para melhorar o desempenho de uma organização, que, normalmente, é reconhecida como a melhor pelas organizações pares.

Bug: Vide *defeito*.

C

Caminho: Uma seqüência de eventos, como por exemplo, comandos executáveis, de um sistema que vai de um ponto de entrada até um ponto de saída.

Caminho do fluxo de controle: Vide *caminho*.

Caminho impossível: Um caminho que não pode ser exercitado por nenhum dos conjuntos de valores de entrada possíveis.

Caminho possível: Um caminho que pode ser exercitado por algum dos conjuntos de valores de entrada possíveis.

Capacidade de aprendizado: A capacidade que o sistema tem de permitir que o usuário aprenda sua utilização. [ISO 9126] Veja também *usabilidade*.

Capacidade de recuperação: A capacidade que um sistema tem de restabelecer um determinado nível de desempenho e recuperar os dados afetados diretamente caso ocorra uma falha. [ISO 9126]. Veja também *confiabilidade*.

Capacidade de reprodução do teste: Atributo de teste que indica se o mesmo resultado é produzido toda vez que o teste é executado.

Capacidade de substituição: A capacidade que um sistema tem de ser usado no lugar de outro para o mesmo objetivo e no mesmo ambiente. [ISO 9126]. Veja também *portabilidade*.

Característica: Atributo de um sistema especificado ou concluído a partir da documentação de requisitos (por exemplo, confiabilidade, usabilidade ou restrições do projeto). [Após IEEE 1008].

Característica da qualidade: Vide *atributo de qualidade*.

Característica da qualidade de software: Vide *atributo de qualidade*.

Característica de software: Vide *característica.*

CASE: Acrônimo para Engenharia de Software Assistida por Computador (do inglês: *Computer Aided Software Engineering*).

Caso de teste: Um conjunto de valores de entrada, pré-condições e pós-condições de execução, e resultados esperados, desenvolvidos para um objetivo particular ou condição de teste, tais como para exercitar um

caminho em particular em um sistema ou para verificar a conformidade a um requisito específico. [Após IEEE 610].

Caso de teste abstrato: Vide *caso de teste de alto nível*.

Caso de teste bloqueado: Um caso de teste que não pode ser executado porque as pré-condições para sua execução não foram preenchidas.

Caso de teste concreto: Vide *caso de teste de baixo nível.*

Caso de teste de alto nível: Caso de teste sem valores concretos (nível de implementação) para os dados de entrada e resultados esperados. São usados operadores lógicos, pois as instâncias para os valores reais ainda não estão definidas e/ou disponíveis. Veja também *caso de teste de baixo nível.*

Caso de teste de baixo nível: Caso de teste com valores concretos (nível de implementação) para os dados de entrada e resultados esperados. Os operadores lógicos dos casos de teste de alto nível são substituídos pelos valores reais que correspondem aos seus objetivos. Veja também *caso de teste de alto nível*.

Caso de teste lógico: Vide *caso de teste de alto nível*.

Caso de uso: Uma seqüência de transações entre um usuário e o sistema com um resultado tangível.

CAST: Acrônimo para Teste de Software Assistido por Computador (do inglês: *Computer Aided Software Testing*). Veja também *automação dos testes*.

Causa raiz: Um acontecimento subjacente que causou uma não conformidade e que deve ser permanentemente eliminado através da melhoria do processo.

Cenário de teste: vide *especificação de procedimento de teste*.

Certificação: O processo de confirmação de que um componente, sistema ou pessoa está conforme com os requisitos especificados, como, por exemplo, passando em um exame.

Ciclo de teste: Execução do processo de teste para cada liberação identificável de um objeto em teste.

Classe de equivalência: Vide *partição de equivalência*.

Cobertura: O grau, expresso em porcentagem, que especifica qual a cobertura dos itens que foram exercitados pela suíte de testes.

Cobertura de caminhos: Porcentagem de caminhos que foram exercitados por uma suíte de testes. 100% da cobertura de caminhos implica em 100% da cobertura de LCSAJs.

Cobertura de condições de decisão: Porcentagem de todas as conseqüências das condições e decisões que foram exercitadas pela suíte de testes. 100% da cobertura de condições de decisão implicam em 100% da cobertura de condições e 100% da cobertura de decisões.

Cobertura de código: Método de análise que determina quais as partes do sistema que foram executadas (cobertas) pela suíte de testes e quais as que não foram executadas, como por exemplo, cobertura de comandos, decisões ou condições.

Cobertura de comandos: Porcentagem de comandos executáveis que foram exercitados pela suíte de teste.

Cobertura de combinações de condições: Vide *cobertura de condições múltiplas.*

Cobertura de combinações de condições dos ramos: Vide *cobertura de condições múltiplas.*

Cobertura de condições: Porcentagem de conseqüências das condições que foram exercitadas pela suíte de testes. 100% da cobertura de condições exige que cada uma das condições seja testada como Verdadeira e Falsa para todos os comandos de decisão.

Cobertura de determinação de condições: Porcentagem de todas as conseqüências das condições que afetam independentemente a conseqüência da decisão que foi exercitada pela suíte de testes. 100% cobertura de determinação de condições implica em 100% da cobertura de condições de decisão.

Cobertura de condições dos ramos: Vide *cobertura de condições.*

Cobertura de condições múltiplas: Porcentagem de combinações de conseqüências de cada condição dentro de um comando que foi exercitado

pela suíte de teste. 100% da cobertura de condições múltiplas implica em 100% da cobertura de determinação de condições.

Cobertura modificada de condições múltiplas: Vide *cobertura de determinação de condições.*

Cobertura de decisões: Porcentagem das conseqüências de decisões que foram exercitadas pela suíte de testes. 100% da cobertura de decisões implicam em 100% da cobertura de ramos e 100% da cobertura de comandos.

Cobertura modificada de decisões da condição: Vide *cobertura de determinação de condições.*

Cobertura de fluxo de dados: Porcentagem de variáveis que foram exercitados pela suíte de testes.

Cobertura de LCSAJ: Porcentagem de LCSAJs de um sistema que foram exercitados pela suíte de testes. 100% da cobertura de LCSAJs implica em 100% da cobertura de decisões.

Cobertura de partição de equivalência: Porcentagem de partições de equivalência que foram exercitadas pela suíte de testes.

Cobertura de ramos: Porcentagem de ramos que foram exercitados pela suíte de testes. 100% da cobertura de ramos implica em 100% da cobertura de decisões e 100% da cobertura de comandos.

Cobertura de valores limite: Porcentagem dos valores limite que foram exercitados pela suíte de testes.

Cobertura de testes: Vide *cobertura*.

Cobertura estrutural: Medidas de cobertura baseadas na estrutura interna do sistema.

Cobertura N-switch: Porcentagem das seqüências das N+1 transições que foram exercitadas pela suíte de testes. [Chow]

Código: Instruções de computador e definições dos dados expressos em uma linguagem de programação ou como saída de um assembler, compilador ou outro tradutor. [IEEE 610]

Código inalcançável: Código que não pode ser alcançado e, portanto não pode ser executado.

Código morto: Vide *código inalcançável.*

Co-existência: A capacidade que o sistema tem de co-existir com outros softwares independentes em um ambiente comum e compartilhando recursos. [ISO 9126]. Veja também *portabilidade.*

Comando: Entidade em uma linguagem de programação, que é, normalmente, a menor unidade indivisível de execução.

Comando executável: Comando que, quando copilado, é traduzido para um código de objeto que será executado continuamente quando o programa for executado; pode agir nos dados.

Comando fonte: vide *comando*.

Comparação dinâmica: Comparação dos resultados obtidos e esperados, que ocorre enquanto o software está sendo executado, como, por exemplo, por uma ferramenta de execução de teste.

Comparação pós-execução: Comparação entre os resultados obtidos e esperados, feita depois da execução do software.

Comparação dos testes: Processo de identificar diferenças entre resultados obtidos e os resultados esperados. A comparação pode ser feita durante a execução dos testes (comparação dinâmica) ou após a execução dos testes.

Comparador: Vide *comparador de teste*.

Comparador de teste: Ferramenta de teste que executa comparações automatizadas de teste.

Compilador: Ferramenta de software que traduz programas escritos em linguagens de alto nível para a sua linguagem de máquina equivalente. [IEEE 610]

Complacência: A capacidade que o sistema tem de ser aderente aos padrões, convenções, leis ou prescrições similares. [ISO 9126]

Complexidade: O grau com que um sistema foi desenhado e/ou como sua estrutura interna é difícil de entender, manter e verificar. Veja também c*omplexidade ciclomática.*

Complexidade ciclomática: Número de caminhos independentes em um programa. [Após McCabe]

Componente: A menor parte do sistema que pode ser testada isoladamente.

Comportamento: A resposta do sistema para um conjunto de pré-condições e valores de entradas.

Comportamento no tempo: Vide *desempenho*

Condição: Expressão lógica que pode ser avaliada como Verdadeira ou Falsa, como por exemplo, A>B. Veja também *condição de teste.*

Condição composta: Duas ou mais condições intercaladas por um operador lógico (E, OU ou E/OU), como por exemplo: "A>B E C>1000".

Condição de ramos: Vide *condição*.

Condição de teste: Um item ou evento de um sistema que pode ser verificado por um ou mais casos de testes, como por exemplo, função, transação, característica, atributo de qualidade ou elemento estrutural.

Condição múltipla: Vide *condição composta*

Confiabilidade: Habilidade do sistema em executar suas funções sob determinadas circunstâncias por um número de operações ou período de tempo determinado. [ISO 9126]

Configuração: Composição de um sistema definido de acordo com o número, natureza e interconexões das suas partes constituintes.

Conjunto de teste: Vide *suíte de teste*

Conjunto de testes base: um conjunto de casos de testes derivados da estrutura interna de um sistema ou especificação para garantir que 100% dos critérios de cobertura especificados serão atingidos.

Conseqüência: Vide *resultado*.

Conseqüência das condições: Avaliação das condições como Verdadeiro ou Falso.

Conseqüência das decisões: O resultado de uma decisão (conseqüentemente, o que determina o ramo a ser utilizado).

Conseqüência dos testes: Vide *resultado*.

Conseqüência esperada: Vide *resultado esperado*

Conseqüência obtida: Vide *resultado obtido.*

Conseqüência prognosticada: Vide *resultado esperado*

Consistência: Grau de uniformidade, padronização e não-contradição entre documentos ou partes de um sistema. [IEEE 610]

Construção diária: Atividade de desenvolvimento onde um sistema completo é copilado e *linkado* a cada dia (normalmente à noite), assim, é disponibilizado um sistema consistente incluindo as últimas modificações.

Contabilidade do status: Elemento de gerenciamento de configuração que consiste em registrar e relatar as informações necessárias para gerenciar efetivamente a configuração. Estas informações incluem uma lista de identificação das configurações aprovadas, o status das alterações propostas para a configuração e os status da implementação das alterações aprovadas. [IEEE 610].

Controle de alteração: Vide *controle de configuração.*

Controle de configuração: Elemento do gerenciamento de configuração, que consiste em avaliar, coordenar, aprovar ou não, e implementar as alterações nos itens de configuração depois que a identificação da configuração é determinada formalmente. [IEEE 610]

Controle de riscos: O processo através do qual são tomadas decisões e implementadas medidas de proteção para reduzir os riscos, ou mantê-los dentro, de níveis especificados.

Controle de versão: Vide *controle de configuração*.

Controle do teste: Uma tarefa de gerenciamento de teste que trata do desenvolvimento e da aplicação de um conjunto de ações corretivas para

manter o projeto nos trilhos quando a monitoração mostra um desvio do que foi planejado. Veja também *gerenciamento de testes*.

COTS: Acrônimo para software de prateleira comercial (do inglês *Commercial Off-The-Shelf software*). Vide *software de prateleira*.

Critério de aceite: O critério de saída que um sistema deve satisfazer para que possa ser aceito por um usuário, cliente, ou entidade autorizada. [IEEE 610]

Critério de completude: Vide *critério de saída*.

Critério de entrada: Conjunto de condições genéricas e específicas que permitem ao processo prosseguir com uma tarefa definida, como por exemplo, fase de teste. O objetivo do critério de entrada é impedir que a tarefa comece requerendo mais esforço (desperdiçado) comparado com o esforço necessário para remover um critério que falhou. [Gilb and Graham]

Critério de finalização dos testes: vide *critério de saída*.

Critério de passou/falhou: Regras de decisão usadas para determinar se um item em teste (função) ou característica passou ou falhou no teste. [IEEE 829]

Critério de reínicio: Atividades de teste que devem ser repetidas quando o teste é reiniciado após uma suspensão. [Após IEEE 829]

Critério de saída: Um conjunto de condições genéricas e específicas, acordados com a parte interessada, para permitir que o processo seja oficialmente completado. O objetivo do critério de saída é evitar que a tarefa seja considerada completa quando ainda existem partes desta tarefa que não foram finalizadas. O critério de saída é usado para reportar e para planejar quando parar de testar. [Após Gilb and Graham]

Critério de suspensão: Critério usado para parar (temporariamente) todas, ou uma parte, das atividades e os itens em teste. [Após IEEE 829]

Cronograma da execução dos testes: Um esquema para a execução de procedimentos de teste. Os procedimentos de teste são incluídos no cronograma de execução dos testes de acordo com seu contexto e a ordem de execução.

D

Dados de teste: Dados que existem (por exemplo, em uma base de dados) antes de um teste ser executado, e que afeta ou é afetado pelo sistema em teste.

Debugar*:* Processo de encontrar, analisar e remover as causas das falhas em um software.

Debugger: Vide *ferramenta para debugar.*

Decisão: Ponto no programa onde o fluxo de controle tem dois ou mais caminhos alternativos. Um nó com duas ou mais arestas para separar os ramos.

Defeito: Problema no sistema que pode levá-lo a falhar durante a execução da funcionalidade solicitada (por exemplo, um comando ou definição de dados incorretos). O defeito, se encontrado durante a execução, pode causar uma falha no sistema.

Definição de dados: Comando executável onde um valor é atribuído a uma variável.

Densidade de defeitos: O número de defeitos identificados em um sistema dividido pelo tamanho do sistema (expresso em termos de métricas padronizadas, como por exemplo, linhas de códigos, número de classes ou pontos por função).

Densidade de falhas: Vide *densidade de defeitos*

Desempenho: O grau com que um sistema realiza suas funcionalidades dentro de determinadas restrições considerando o tempo de processamento e a taxa de envio de dados. [Após IEEE 610] Veja também *eficiência*.

Desenvolvimento dirigido pelo teste: (Do inglês *Test-Driven Development* – TDD). Uma forma de desenvolvimento de software onde os casos de teste são desenvolvidos, e freqüentemente automatizados, antes do software ser desenvolvido.

Desvio: Vide *incidente.*

Diagrama de estados: Diagrama que define os estados que um sistema pode assumir, e mostra os eventos ou circunstâncias que causam e/ou resultam de uma mudança de um estado para outro. [IEEE 610]

Disponibilidade: Grau com que um sistema é operacional e acessível quando requisitado para uso. Usado em porcentagem, normalmente. [IEEE 610]

Domínio: Conjunto de onde podem ser selecionados valores válidos para as entradas e/ou saídas.

Domínio de entrada: Conjunto de onde podem ser selecionados valores válidos para as entradas. Veja também *domínio.*

Domínio de saída: Conjunto de onde podem ser selecionados valores válidos para as saídas. Veja também *domínio*.

Driver: Componente de software ou ferramenta de teste que substitui o componente que controla e/ou chama o sistema que está sendo testado. [Após TMap]

***Driver* de teste:** Vide *driver*.

E

Efeito da investigação: Efeito no sistema causado quando ele está sendo avaliado por um instrumento de mensuração que pode ser, por exemplo, uma ferramenta de teste de desempenho ou monitoração. Por exemplo, o desempenho pode ser ligeiramente pior quando estão sendo usadas ferramentas de teste de desempenho.

Eficiência: Capacidade que o sistema tem de fornecer o desempenho apropriado, relativo à quantidade de recursos utilizados sob as condições declaradas [ISO 9126].

Emulador: Um artifício, programa ou sistema que aceita as mesmas entradas e produz as mesmas saídas que um dado sistema. [IEEE 610]. Veja também *simulador*.

Encerramento do teste: Durante a fase de encerramento do teste, são coletados os dados do processo de teste e das atividades que foram finalizadas, para consolidar a experiência, o testware, os fatos e os números. A fase de encerramento do teste consiste na finalização e

arquivamento do *testware* e avaliação do processo de teste incluindo a preparação do relatório de avaliação. Veja também *processo de teste*

Engano: Vide *erro*.

Entrada: Variável (armazenada dentro de um componente ou fora dele) que é lida por um componente.

Entrada especificada: Entrada para a qual a especificação prevê um resultado.

Entrada de teste: Dados que o objeto em teste recebe de uma fonte externa, durante a execução dos testes. Essa fonte pode ser hardware, software ou humana.

Entregável: Qualquer produto que deve ser entregue para alguém que não seja seu autor.

Erro: Uma ação humana que produz um resultado incorreto. [Após IEEE 610]

Error guessing: Técnica de projeto de teste, onde a experiência do testador é usada para antecipar quais os defeitos, resultantes de erros cometidos, podem estar presentes no sistema em teste e para planejar testes específicos para encontrá-los.

Escala da medida: Escala que restringe o tipo de análise de dados que pode ser executada. [ISO 14598]

Escalabilidade: A capacidade que um sistema tem ser atualizável para aceitar cargas maiores. [Após Gerrard]

Escape da memória: Defeito na lógica da alocação dinâmica de memória em um programa que faz com ele não libere a memória depois que terminou de usar, causando, eventualmente, uma falha no programa devido à falta da memória.

Escritura de teste: Declaração dos objetivos do teste, e possivelmente, idéias de como testar. A escritura de teste é usada com freqüência para os testes exploratórios. Veja também *teste exploratório.*

Escrivão: Pessoa que registra, em um formulário, cada falha encontrada e sugestões para a melhoria do processo durante uma reunião de revisão. Ele deve garantir que o formulário de registro é legível e entendível.

Especificação: Documento que especifica requisitos, projeto, comportamento e outras características de um sistema, além de freqüentemente especificar procedimentos para determinar se eles foram satisfeitos. Idealmente a especificação deve ser completa, precisa e testável. [Após IEEE 610]

Especificação de casos de teste: Documento que especifica um conjunto de casos de teste (objetivo, entradas, ações de testes, resultados esperados e pré-condições de execução) para um item em teste. [Após IEEE 829].

Especificação de componente: Descrição da funcionalidade de um componente, em termos de valores de saída, para os valores de entrada especificados, e do comportamento não-funcional requerido (por exemplo: utilização de recursos).

Especificação de procedimento de teste: Documento que especifica a seqüência de ações para a execução dos testes. Conhecido também como roteiro de teste ou roteiro manual de teste. [Após IEEE 829].

Especificação de projetos de teste: Documento que especifica as condições de testes (itens de cobertura) para um item em teste, a técnica de teste detalhada e identifica os casos de teste de alto nível associados. [Após IEEE 829].

Especificação de teste: Documento que consiste em uma especificação de projeto de teste, especificação de casos de testes, e/ou especificação de procedimentos de teste.

Estabilidade: A habilidade de um sistema em evitar efeitos inesperados devido às modificações no software. [ISO 9126] Vide *manutenibilidade*.

Estratégia de teste: Descrição de alto-nível dos níveis de testes, com os respectivos testes, que serão executados para uma organização ou programa (um ou mais projetos).

Estrutura de teste: Ambiente de teste composto de *drivers* e *stubs* necessários para executar o teste.

Etapas de teste: Vide *nível de teste*.

Exatidão: A capacidade que o sistema tem de fornecer efeitos (ou resultados) corretos (ou acordados) com o grau de precisão necessário.

[ISO 9126]. Veja também *teste de funcionalidade*.

Execução do teste: (1) Processo de testar um sistema, produzindo resultados (resultado obtido). (2) Execução de testes em uma versão específica do objeto em teste.

Exercitado: Um elemento do programa é exercitado por um caso de teste quando o valor de entrada, como, por exemplo, um comando, uma decisão ou outro elemento estrutural, leva à execução desse elemento.

F

Faixa larga delphi: Uma técnica especializada de estimativa de teste que visa realizar uma estimativa precisa usando o conhecimento coletivo dos membros da equipe.

Falha: (1) Uma falha ocorre quando o resultado obtido é diferente do resultado esperado. (2) um sistema desvia do resultado, serviço ou entregável esperado. [Após Fenton].

Falha no teste: Vide *falha*.

Falta: Vide *defeito*

Fase de execução dos testes: Período de tempo no ciclo de desenvolvimento do software durante o qual os componentes de software são executados e avaliados para determinar se os requisitos foram satisfeitos ou não. [IEEE 610]

Fase de requisitos: Período de tempo no ciclo de vida do software durante o qual os requisitos de um sistema são definidos e documentados. [IEEE 610]

Fase de teste: Um conjunto distinto de atividades de testes armazenadas em uma fase gerenciável do projeto. [Após Gerrard].

Ferramenta de acompanhamento de defeito: vide *ferramenta de gerenciamento de defeitos.*

Ferramenta de análise dinâmica: Ferramenta que fornece informações em tempo de execução sobre o estado do código. Essas ferramentas são usadas, por exemplo, para identificar ponteiros sem atribuições, verificar

a aritmética dos ponteiros, monitorar a alocação, uso e desalocação de memória e para sinalizar escapes de memória.

Ferramenta de análise estática: Vide *analisador estático*.

Ferramenta de captura/re-execução: Vide *ferramenta de captura/reprodução.*

Ferramenta de captura/reprodução: Ferramenta de execução de testes onde as entradas são gravadas durante o teste manual para gerar scripts de testes automatizados que podem ser re-executados depois. Estas ferramentas são usadas, normalmente, para apoiar os testes regressivos automatizados.

Ferramenta de cobertura: Ferramenta que fornece métricas objetivas sobre quais elementos estruturais, como por exemplo, comandos e ramos, foram exercitados pela suíte de testes.

Ferramenta de execução de testes: Ferramenta de teste capaz de executar outro software usando um script de teste automatizado, como por exemplo, as ferramentas de captura/reprodução. [Fewster and Graham]

Ferramenta de gerenciamento de configuração: Ferramenta que apóia a identificação e controle dos itens de configuração, seus status baseado nas suas mudanças e versões, e liberação de *baselines* compreendendo os itens de configuração.

Ferramenta de gerenciamento de defeitos: Ferramenta que facilita o registro e o acompanhamento do *status* dos defeitos. Normalmente essas ferramentas são orientadas a *workflow* e permitem rastrear e controlar a alocação, correção e re-teste dos defeitos e fornecer facilidades de reporte. Veja também *ferramenta de gerenciamento de incidentes*.

Ferramenta de gerenciamento de incidentes: Ferramenta que facilita o registro e o acompanhamento do *status* dos incidentes. Normalmente essas ferramentas são orientadas a *workflow* e tem vantagens para rastrear e controlar a alocação, correção e re-teste dos incidentes, e fornecer facilidades de reporte. Veja também *ferramenta de gerenciamento de defeitos*.

Ferramenta de gerenciamento de requisitos: Ferramenta que apóia o registro dos requisitos, seus atributos (como por exemplo, prioridade e responsável pelo conhecimento) e anotações. Facilita a rastreabilidade e o

gerenciamento de alterações nos requisitos. Algumas ferramentas de gerenciamento de requisitos também fornecem facilidades para análises estáticas, como verificação de consistência e violações da regras pré-definidas para os requisitos.

Ferramenta de gerenciamento de testes: Ferramenta que apóia o gerenciamento de testes e controla parte do processo de teste. Possui facilidades, tais como gerenciamento do *testware*, cronograma dos testes, registro de resultados, acompanhamento do progresso, gerenciamento de incidentes e relatório dos testes.

Ferramenta de monitoração: Vide *monitor*

Ferramenta de preparação de dados de teste: Ferramenta que habilita os dados para serem selecionados, criados, gerados, manipulados e editados em uma base de dados existente, para que possam ser usados nos testes.

Ferramenta de projeto de teste: Ferramenta que apóia a atividade de projeto de teste gerando as entradas de teste a partir de uma especificação (que pode ser guardada em um repositório de uma ferramenta CASE, como por exemplo, uma ferramenta de gerenciamento de requisitos), a partir de condições de teste especificadas (guardadas na própria ferramenta), ou a partir do código.

Ferramenta de registro/reprodução: Vide *ferramenta de captura/reprodução*.

Ferramenta de revisão: Ferramenta que apóia o processo de revisão. Suas características básicas são apoiar o planejamento e acompanhamento da revisão, apoiar a comunicação, permitir revisões colaborativas, e oferecer um repositório para coleta e reporte de métricas.

Ferramenta de segurança: Ferramenta que apóia a segurança operacional.

Ferramenta de teste: Ferramenta que apóia uma ou mais atividades de teste, tais como planejamento e controle, especificação, construção dos arquivos e dados iniciais, execução e análise do teste. [TMap] Veja também *CAST*.

Ferramenta de teste de desempenho: Ferramenta que apóia os testes de desempenho e que, normalmente, possui duas facilidades: geração de carga e mensuração das transações do teste. A geração de carga pode

simular múltiplos usuários ou grandes volumes de dados de entrada. Durante a execução, os tempos de resposta para determinadas transações são medidos e registrados. As ferramentas de teste de desempenho, normalmente, fornecem relatórios baseados nos registros dos testes e gráficos da carga em função dos tempos de resposta.

Ferramenta de teste de segurança: Ferramenta que apóia os testes das características e vulnerabilidades de segurança.

Ferramenta para *debugar*: Ferramenta usada por programadores para reproduzir falhas, investigar o estado dos programas e encontrar o defeito correspondente. Essas ferramentas permitem que os programadores executem os programas passo a passo, parem o programa em qualquer comando, e atribuam valores e examinem as variáveis dos programas.

Fluxo de controle: Seqüência de eventos (caminhos) na execução de um sistema.

Fluxo de dados: Representação abstrata de uma seqüência e possíveis mudanças de estado dos objetos de dados, onde o estado de um objeto pode ser: criação, uso ou destruição. [Beizer]

Funcionalidade: A capacidade que o sistema tem em fornecer funções que satisfaçam as necessidades declaradas e implícitas quando o software é executado. [ISO 9126]

G

Garantia de qualidade: Parte do gerenciamento da qualidade focado em fornecer a garantia de que os requisitos de qualidade serão satisfeitos. [ISO 9000]

Gerador de teste: Vide *ferramenta de preparação de dados de teste.*

Gerenciamento da qualidade: Atividades coordenadas para direcionar e controlar uma organização levando em consideração a qualidade. Isso geralmente inclui o estabelecimento de políticas e objetivos de qualidade, planejamento, controle, garantia e melhoria da qualidade. [ISO 9000]

Gerenciamento de configuração: Disciplina que aplica orientação e supervisão técnica e administrativa para: identificar e documentar as características funcionais e físicas de um item de configuração, controlar alterações nessas características, gravar e relatar o processamento da

alteração e o status da implementação, e verificar a conformidade com os requisitos especificados. [IEEE 610]

Gerenciamento de defeitos: Processo de reconhecimento, investigação, ação e ordenação dos defeitos. Isso envolve registrar os defeitos, classificá-los e identificar seu o impacto. [Após IEEE 1044]

Gerenciamento de incidentes: Processo de reconhecimento, investigação, ação e ordenação dos incidentes. Esse processo visa registrar os incidentes, classificá-los e identificar seu o impacto. [Após IEEE 1044]

Gerenciamento de problemas: vide *gerenciamento de defeitos.*

Gerenciamento de riscos: Aplicação sistemática de procedimentos e práticas para identificar, analisar, priorizar e controlar os riscos.

Gerenciamento de testes: Planejamento, estimativa, monitoração e controle das atividades de teste, normalmente realizado pelo gerente de teste.

Gerente de teste: Pessoa responsável pelo gerenciamento das atividades e recursos dos testes. É ele quem direciona, controla, administra, planeja e regula a avaliação dos objetos em teste.

Grafo causa-efeito: Representação gráfica das entradas e/ou estímulos (causas) e suas saídas (efeitos) associadas. Esse grafo pode ser usado para definir os casos de testes.

Grafo de fluxo de controle: Seqüência dos eventos (caminhos) na execução de um sistema.

Gravação de teste: Vide *log do teste*.

Gravar o teste: Vide *registrar o teste*.

Grupo de controle de alteração: Vide *grupo de controle de configuração.*

Grupo de controle de configuração: Grupo de pessoas responsáveis pela avaliação e aprovação (ou não aprovação) das alterações propostas nos itens de configuração. Além disso, eles devem garantir a implementação das alterações aprovadas. [IEEE 610]

Guia de instalação: Instruções, fornecidas em mídia apropriada, que guiam o processo de instalação. Pode ser um guia manual, um procedimento passo a passo, um tutorial de instalação ou qualquer outra descrição de processo similar.

H

I

Identificação de configuração: Elemento do gerenciamento de configuração, que consiste em selecionar os itens de configuração para um sistema e gravar suas características funcionais e físicas em um documento técnico. [IEEE 610]

Identificação de riscos: Processo de identificar riscos usando técnicas como *brainstorming*, *checklists* e históricos de falhas.

Incidente: Qualquer evento que necessita de investigação quando ocorre. [Após IEEE 1008]

Incidente de teste: Vide *incidente*.

Incidente de teste de software: Vide *incidente*.

Independência: Separação das responsabilidades para incentivar a realização de testes objetivos. [Após DO-178b]

Indicador de desempenho: Métrica de efetividade e/ou eficiência usada para guiar e controlar o desenvolvimento progressivo, como, por exemplo, lidar com a falta de tempo para o desenvolvimento do software. [CMMI]

Indicador de desempenho chave: Vide *indicador de desempenho*.

Indicador de desempenho do teste: Uma métrica de efetividade e/ou eficiência usada para guiar e controlar o desenvolvimento progressivo dos testes, como por exemplo, a porcentagem detecção de defeitos.

Infraestrutura de teste: Artefatos necessários para executar os testes. Composto de ambiente e ferramenta de testes e procedimentos e ambientes da organização.

Inspeção: Um tipo de revisão por pares que confia no exame visual dos documentos para detectar os defeitos, como por exemplo, violações dos padrões de desenvolvimento e não-conformidade à documentação de alto-nível. Por ser uma técnica de revisão formal é sempre baseada em um procedimento documentado. [Após IEEE 610, IEEE 1028] Veja também *revisão por pares.*

Inspetor: Vide *revisor*.

Instalabilidade: A capacidade que o sistema tem de ser instalado em um ambiente especificado [ISO 9126] Veja também *portabilidade*.

Instrumentação: Inserção de código adicional no programa para coletar informações sobre o comportamento do programa durante a execução, por exemplo, para medir a cobertura de código.

Instrumentador: Ferramenta utilizada para realizar a instrumentação.

Instrumentador do programa: Vide *instrumentador*.

Integração: O processo de combinar componentes ou sistemas em partes maiores.

Integração funcional: Técnica de integração que combina os componentes para obter antecipadamente uma funcionalidade básica que seja executável. Veja também *teste de integração*.

Interoperabilidade: A capacidade que o sistema tem de interagir com o um ou mais sistemas especificados. [Após ISO 9126] Veja também *funcionalidade*.

Inventário da Mensuração da Usabilidade do Software (SUMI): (Do inglês *Software Usability Measurement Inventory*). Técnica de teste de usabilidade baseada em questionários para avaliar a usabilidade (como por exemplo, a satisfação do usuário) de um sistema. [Veenendaal]

Item de cobertura: Entidade ou propriedade usada como base para a cobertura dos testes, como por exemplo, partição de equivalência ou comandos de código.

Item de configuração: Agregação de hardware, ou software, ou ambos, para o gerenciamento da configuração. São tratados como uma entidade única no processo de gerenciamento de configuração. [IEEE 610]

Item em teste: Elemento que será testado. Há geralmente um objeto em teste e muitos itens em teste. Veja também *objeto em teste*.

K

L

LCSAJ: Acrônimo para Seqüência linear e salto no código. (Do inglês *Linear Code Sequence and Jump*). Consiste em três itens (por convenção, identificados pelo número da linha na listagem do código fonte): (i) o início da seqüência linear de comandos executáveis, (ii) o fim da seqüência linear, e (iii) a linha alvo, para a qual o fluxo de controle é transferido no fim da seqüência linear.

Legibilidade: A habilidade do sistema em permitir que o usuário compreenda se o software é apropriado e como ele pode ser usado em tarefas e condições particulares do uso. [ISO 9126] Veja também *usabilidade*

Líder de inspeção: Vide *moderador.*

Líder de teste: Vide *gerente de teste*.

Linguagem de *script*: Linguagem de programação usada para escrever *scripts* de testes executáveis. Essas linguagens são usadas por ferramentas de execução de teste, como por exemplo, em ferramentas de captura/reprodução.

Log do teste: Registro cronológico dos detalhes relevantes da execução de testes. [IEEE 829].

M

Manutenção: Alteração de um sistema, após a entrega, para corrigir defeitos, melhorar o desempenho ou outros atributos, ou adaptar o produto a um ambiente modificado. [IEEE 1219]

Manutenibilidade: A facilidade com que um sistema pode ser alterado para corrigir defeitos, satisfazer novos requisitos, facilitar manutenções futuras ou ser adaptado para um ambiente modificado. [ISO 9126]

Marco: Marco em um projeto. Ponto onde os entregáveis e resultados (intermediários) devem estar prontos.

Máscara de defeito: Ocorrência onde um defeito impede a detecção de outro. [Após IEEE 610]

Máscara de falha: Vide *máscara de defeito*

Máquina finita de estados: Modelo computacional que consiste em um número finito de estados e transições entre os estados. Essa máquina possibilita o acompanhamento de ações. [IEEE 610]

Maturidade: (1) A capacidade de uma organização em relação à efetividade e eficiência de seus processos e práticas de trabalho. Veja também *modelo de maturidade de capacidade, modelo de maturidade de teste*. (2) A capacidade que o sistema tem de evitar falhas como conseqüência dos defeitos no software. [ISO 9126] Veja também *confiabilidade*.

Medida: Número ou categoria atribuído a uma entidade através de uma mensuração. [ISO 14598]

Melhoria do Processo de Teste (TPI): (do inglês: *Test Process Improvement*) Modelo para melhoria contínua do processo de teste que descreve os elementos chave de um processo de testes efetivo, especialmente para testes de sistemas e de aceite.

Mensurar: Processo de atribuir um número ou uma categoria a uma entidade para descrever um atributo dessa entidade. [ISO 14598]

Método de árvore de classificação: Técnica de projeto de teste caixa-preta onde os casos de teste são projetados para executar combinações dos representantes dos domínios de entrada e/ou de saída. Os casos de teste são descritos usando uma árvore de classificação. [Grochtmann]

Métrica: Uma escala de medida e o método usado para medí-la. [ISO 14598]

Métricas de cobertura de Chow: Vide *cobertura N-switch*. [Chow]

Mitigação de riscos: Vide *controle de riscos*.

Modelo de desenvolvimento incremental: Modelo de desenvolvimento onde um projeto é quebrado em uma série de incrementos. Cada

incremento é uma parte da funcionalidade dos requisitos totais do projeto. Os requisitos são priorizados e entregues na ordem da prioridade para o incremento apropriado. Em algumas (mas não em todas) versões deste modelo de ciclo de vida, cada subprojeto segue "mini modelo V" com suas próprias fases de projeto, codificação e testes.

Modelo de desenvolvimento iterativo: Modelo de desenvolvimento onde um projeto é quebrado em um número, geralmente, grande de iterações. Uma iteração é um ciclo de desenvolvimento completo que resulta na entrega (interna ou externa) de um produto executável, que é um subconjunto do produto final em desenvolvimento. O produto entregue aumenta a cada iteração para se transformar no produto final.

Modelo de densenvolvimento "V": Modelo de desenvolvimento que descreve as atividades do ciclo de vida do desenvolvimento do software desde a especificação de requisitos até a implantação. O modelo "V" ilustra como as atividades de teste podem ser integradas a cada fase do desenvolvimento do software.

Modelo de Maturidade de Capacidade (CMM): (Do inglês *Capability Maturity Model*) Modelo de cinco níveis que descreve os elementos chave de um processo de software efetivo. O CMM possui boas práticas para planejamento, engenharia e gerenciamento do desenvolvimento, e manutenção de software. [CMM]

Modelo de Maturidade da Capacidade Integrado (CMMI): (Do inglês *Capability Maturity Model Integration*). Modelo que descreve os elementos chave de um processo de desenvolvimento e manutenção efetivos. O CMMI cobre boas práticas para planejamento, engenharia e gerenciamento do desenvolvimento e manutenção do produto. O CMMI é o sucessor natural do CMM. [CMMI]

Modelo de Maturidade de Teste (TMM): (Do inglês: *Test Maturity Model*). Modelo de 5 níveis para a melhoria do processo de testes. Está relacionado ao CMM, que descreve os elementos chave de um processo de testes efetivo.

Moderador: Líder e principal responsável por uma inspeção ou o outro processo da revisão.

Modo de falha: Manifestação física ou funcional de uma falha. Um sistema em modo de falha pode ser caracterizado, por exemplo, por operação lenta, saídas incorretas, ou execução interrompida. [IEEE 610]

Modo de Falha e Análise do Efeito (FMEA): (Do inglês *Failure Mode and Effect Analysis*) Técnica sistemática para identificar os riscos, analisar e identificar os possíveis modos de falhas e tentar prevenir sua ocorrência.

Módulo: Vide *componente*

Monitor: Ferramenta de software ou dispositivo de hardware executado concorrentemente com o sistema em teste para supervisionar, registrar e/ou analisar seu comportamento. [Após IEEE 610]

Monitoração dos testes: Tarefa de gerenciamento de testes que trata das atividades relacionadas à verificação periódica do status do projeto de teste. Os relatórios comparam o que aconteceu com o que foi planejado. Veja também *gerenciamento de testes*.

N

Não-conformidade: Não satisfação a um requisito especificado. [ISO 9000]

Nível de teste: Conjunto de atividades de teste que são organizadas e gerenciadas em conjunto. Um nível de teste está ligado às responsabilidades em um projeto. Exemplos de níveis de testes são testes de integração, sistemas e aceite. [Após Tmap]

Nota de liberação: Documento que identifica os itens em teste, sua configuração, status atual e outras informações de liberação. Esse documento é entregue pela equipe de desenvolvimento para a equipe de testes, e possivelmente para outros interessados, no início da fase de execução de testes. [Após IEEE 829]

Número ciclomático: Vide *complexidade ciclomática.*

O

Objetivos do teste: Razão ou propósito para planejar e executar um teste.

Objeto em teste: Sistema que será testado. Veja também *item em teste*.

Operabilidade: A capacidade que o sistema tem de permitir que o usuário opere-o e controle-o. [ISO 9126] Veja também *usabilidade*.

Oráculo: Vide *oráculo de teste*.

Oráculo de teste: Fonte para determinar os resultados esperados que serão comparados com os resultados obtidos no sistema em teste. Um oráculo pode ser um sistema existente (um *benchmark*), manual do usuário ou conhecimento especializado de um indivíduo, mas não deve ser o código fonte. [Após Adrion]

P

Par definição-uso: Associa a definição de uma variável ao seu uso. Uma variável pode ser de uso computacional (por exemplo, multiplicação) ou de execução direta de um caminho (uso "predicado").

Partição de equivalência: Uma parte do domínio de entrada ou saída para o qual o comportamento do sistema deve ser o mesmo, de acordo com a especificação.

Particionamento por equivalência: Técnica de projeto de teste caixa-preta onde os casos de teste são projetados para executar dados que representem as partições de equivalência. A princípio, os casos de teste devem cobrir cada partição pelo menos uma vez.

Passar no teste: Vide *Passar*.

Passar: O objeto em teste "passa" no teste quando o resultado obtido é igual ao resultado esperado.

Percentual de Detecção de falhas: Vide *porcentagem de detecção de defeitos*

Planejamento do teste: Atividade de criar ou atualizar um plano de testes.

Plano de teste: Documento que descreve o escopo, técnica, recursos e cronogramas das atividades de testes planejadas. Ele identifica entre outros itens, as características que serão testadas, as tarefas do teste, quem vai executar cada tarefa, o grau de independência do testador, o ambiente de teste, as técnicas de projeto de teste e os critérios de entrada e saída que serão usados, o racional para estas escolhas e o plano de contingência de riscos exigido. Ele é um registro do processo de planejamento de testes. [Após IEEE 829].

Plano de teste do projeto: Vide *plano de teste mestre*.

Plano de teste mestre: Plano de teste que trata, normalmente, de vários níveis de teste. Veja também *plano de teste*.

Plano de teste para a fase: Plano de teste que, normalmente, se refere a uma fase de teste. Veja também *plano de teste*.

Plano de teste para o nível: Plano de teste que trata, normalmente, de um nível de teste. Veja também *plano de teste*.

Política de teste: Documento que descreve os princípios, técnicas e objetivos principais da organização com relação aos testes.

Ponto de entrada: O primeiro comando executável em um sistema.

Ponto de saída: O último comando executável em um sistema.

Porcentagem de Detecção de Defeitos: Número de defeitos encontrados em uma fase de teste dividido pela soma dos defeitos encontrados nesta fase e dos defeitos encontrado nas fases subseqüentes.

Portabilidade: A facilidade com que um sistema pode ser transportado de um ambiente hardware ou software para outro. [ISO 9126]

Pós-condição: Condições de estado e ambiente que devem ser satisfeitas após a execução de um teste ou procedimento de teste.

Pré-condição: Condições de estado e ambiente que devem ser satisfeitas antes que seja executado um teste ou procedimento de teste.

Pré-teste: Vide *teste de entrada.*

Prioridade: Nível de importância (ao negócio) atribuída a um item, como, por exemplo, defeito.

Problema: Vide *defeito*.

Procedimento de teste: Vide *especificação de procedimento de teste*.

Processo: Conjunto de atividades inter-relacionadas que transformam entradas em saídas. [ISO 12207]

Processo de teste: O processo teste básico inclui planejamento, especificação, execução, registro, verificação de completude e atividades de encerramento. [Após BS 7925/2]

Programação em pares: Técnica de desenvolvimento de software onde as linhas de código (produção e/ou teste) de um componente são escritas por dois programadores trabalhando em um mesmo computador. Com isso, implicitamente, é realizada uma revisão de código em tempo real.

Projeto: É um conjunto de atividades coordenadas e controladas, com data de início e fim para atingir um objetivo que deve estar de acordo com requisitos específicos, incluindo restrições de tempo, custo e recursos. [ISO 9000]

Projeto de teste: Vide *especificação de projetos de teste.*

Pseudo-aleatório: Uma seqüência que parece ser aleatória, entretanto é gerada de acordo com uma seqüência pré-definida.

Q

Qualidade: O grau com que um componente, sistema ou processo adere a requisitos especificados e/ou necessidades e expectativas de usuários/clientes. [Após IEEE 610]

Qualidade de *software*: A totalidade de funcionalidades e características de um sistema relacionadas à sua capacidade de satisfazer necessidades explícitas ou implícitas. [Após ISO 9126]

R

Ramo: Bloco que pode ser selecionado para ser executado se a construção tiver um de dois ou mais caminhos alternativos do programa disponíveis, como, por exemplo, *case, jump, go to, if-then-else.*

Rastreabilidade: A habilidade de identificar no software, os itens presentes na documentação e vice-versa (por exemplo, associar os requisitos com seus respectivos testes). Veja também *rastreabilidade horizontal, rastreabilidade vertical*.

Rastreabilidade horizontal: Rastrear os requisitos de um nível de teste na documentação de teste, como, por exemplo, plano de teste,

especificação do projeto do teste, especificação do caso do teste e especificação do procedimento de teste ou do script de teste.

Rastreabilidade vertical: Rastrear os requisitos nas camadas de desenvolvimento, desde a documentação até os componentes.

Redator: Vide *Escrivão.*

Registrar o teste: Processo de registrar as informações sobre testes executados no *log* do teste.

Registro da execução do teste: Vide *log do teste*.

Registro do incidente: Gravação dos detalhes de qualquer incidente ocorrido.

Relatório de avaliação do teste: Documento produzido no final do processo de teste resumindo todas as atividades e resultados do teste. Contém também uma avaliação dos processos de teste e lições aprendidas.

Relatório de *Bugs*: Vide *relatório de defeitos.*

Relatório de defeitos: Documento que descreve qualquer problema que pode causar uma falha ao realizar a função solicitada em um sistema. [Após IEEE 829]

Relatório de desvios: vide *relatório de incidentes.*

Relatório de incidentes: Documento que relata qualquer evento ocorrido que necessite de investigação. [Após IEEE 829]

Relatório de incidentes de teste: Vide *relatório de incidentes*.

Relatório de incidentes de teste de software: Vide *relatório de incidentes*.

Relatório de problemas: Vide *relatório de defeitos.*

Relatório de resumo do teste: Documento que resume as atividades e resultados do teste. Ele contém também a avaliação dos itens em teste em relação aos critérios de saída correspondentes. [Após IEEE 829]

Relatório de teste: Vide *relatório de resumo do teste*.

Relatório de transmissão de itens em teste: Vide *nota de liberação*.

Relatório de transmissão de itens: Vide *nota de liberação*.

Requisito: Condição ou habilidade que um usuário precisa para resolver um problema ou atingir um objetivo. Essa condição deve existir em um sistema, e deve satisfazer um contrato, padrão, especificação, ou outro documento formal usado. [Após IEEE 610]

Requisito de teste: Vide *condição de teste*.

Requisito funcional: Requisito que especifica uma funcionalidade que um sistema deve executar. [IEEE 610]

Requisito não-funcional: Requisito que não trata de funcionalidades e sim de atributos de qualidade tais como confiabilidade, eficiência, usabilidade, manutenibilidade e portabilidade.

Requisitos testáveis: Requisito que permite o estabelecimento do projeto de testes (e casos de teste) e a execução dos testes para determinar se os requisitos foram atendidos. [Após IEEE 610].

Resultado: Conseqüência da execução de um teste, incluindo saídas para tela, alteração nos dados, relatórios e exibição de mensagens de comunicação. Veja também *resultado obtido*, *resultado esperado*.

Resultado do teste: Vide *resultado*.

Resultado esperado: Comportamento previsto pela especificação, ou outra fonte, do sistema sob as condições especificadas.

Resultado obtido: O comportamento produzido/observado quando um sistema é testado.

Revisão: Avaliação de um produto ou status de um projeto para determinar discrepâncias em relação aos resultados planejados e recomendar melhorias. São exemplos de revisão: revisão gerencial, revisão informal, revisão técnica, inspeção, e *walktrough*. [Após IEEE 1028]

Revisão *ad hoc*: Vide *revisão informal*.

Revisão de gerenciamento: Avaliação sistemática do processo de aquisição, fornecimento, desenvolvimento, operação ou manutenção do software. Essa revisão é feita pelo gerente, que monitora o progresso, determina o status dos planos e cronogramas, confirma os requisitos e sua alocação nos sistemas ou avalia a efetividade das técnicas de gerenciamento para atingir o objetivo. [Após IEEE 610, IEEE 1028].

Revisão de testabilidade: Verificação detalhada da base de teste para determinar se a mesma está com um nível de qualidade adequado para servir como entrada para o processo do teste. [Após TMap]

Revisão formal: Revisão caracterizada por requisitos e procedimentos documentados, como, por exemplo, inspeção.

Revisão informal: Revisão que não é baseada em um procedimento formal (documentado).

Revisão por pares: Uma revisão de um sistema por pares do autor do produto com o objetivo de identificar defeitos e melhorias. Exemplos: inspeções, revisões técnicas, e Walkthrough.

Revisão técnica: Um grupo de discussão formado por pares com o objetivo de chegar a um consenso sobre qual técnica usar. [Gilb and Graham, IEEE 1028] Veja também *revisão por pares*.

Revisor: Pessoa envolvida na revisão que identifica e descreve anomalias no produto ou projeto que está sendo revisado. Os revisores podem ser escolhidos para representar diferentes pontos de vista e papéis no processo de revisão.

Reteste: Teste que executa casos de teste que falharam na última vez em que foram executados, para verificar o sucesso das ações de correção.

Risco: Um fator que, no futuro, pode resultar em conseqüências negativas; normalmente expressado como impacto e probabilidade.

Risco do produto: Risco relacionado diretamente ao objeto em teste. Veja também *risco*.

Risco do projeto: Risco relacionado ao gerenciamento e controle do projeto (de teste). Veja também *risco*.

Robustez: O grau de funcionamento correto do sistema na presença de entradas inválidas ou condições de stress do ambiente. [IEEE 610] Veja também *tolerância a erros*, *tolerância a falhas*.

Roteiro de teste: Usado para referenciar uma especificação de procedimento de teste. Veja também *especificação de procedimento de teste*.

S

Saída: Variável (armazenada dentro ou fora de um componente) escrita por um componente.

Script de teste: vide *roteiro de teste*

Segurança: (1) Habilidade do sistema em alcançar níveis aceitáveis de risco em prejudicar as pessoas, o negócio, o software, a propriedade ou o ambiente. [ISO 9126]. (2) Atributos de um sistema relacionados à sua habilidade de prevenir acesso não autorizado a programas e dados sejam eles acidentais ou deliberados. [ISO 9126] Veja também *funcionalidade.*

Semear erros: O processo de adicionar, intencionalmente, defeitos conhecidos aos que já existem no sistema com o objetivo de monitorar a taxa de detecção e remoção e estimar o número de defeitos remanescentes. [IEEE 610]

Sensibilização de caminho: Escolha de conjunto de valores de entrada que força a execução de um determinado caminho.

Severidade: O grau do impacto de um defeito no desenvolvimento ou operação de um sistema. [Após IEEE 610]

Simulação: Representação das características comportamentais selecionadas de um sistema, físico ou abstrato, por outro sistema. [ISO 2382/1]

Simulador: Dispositivo, programa ou sistema usado durante os testes, que se comporta ou opera como outro sistema quando munido de um conjunto de entradas controladas. [Após IEEE 610, DO178b]. Veja também *emulador*.

Sistema: Conjunto de componentes organizados para atender uma função ou conjunto de funções específicas. [IEEE 610]

Situação de teste: Vide *condição de teste*.

***Software*:** Programas de computador, procedimentos, documentações associadas e dados pertinentes à operação de um sistema de computador [IEEE 610]

***Software* de prateleira:** Sistema desenvolvido para o mercado em geral e entregue com formato idêntico para muitos usuários.

***Software* de prateleira comercial:** Vide *software de prateleira.*

***Software* sob medida:** Software desenvolvido para um conjunto de usuários ou clientes em particular. O oposto de software de prateleira.

***Software* padronizado:** Vide *software de prateleira*

Software* personalizado*: Vide *software sob medida*

***Stub*:** Implementação de um componente de software que será usado para desenvolver ou testar um componente que o chama ou é dependente dele. O *stub* substitui o componente chamado. [Após IEEE 610]

Sub-caminho: Seqüência de comandos executáveis dentro de um componente.

Suíte de teste: Conjunto de casos de testes para um sistema em teste, onde a pós-condição de um teste é freqüentemente usada como pré-condição para o próximo teste.

Suíte de casos de teste: vide *suíte de teste*

T

Tabela de decisão: Tabela que exibe as combinações de entradas e/ou estímulos (causas) com suas saídas e/ou ações (efeitos) associadas. Essas combinações podem ser usadas para planejar casos de teste.

Tabela de decisão de causa-efeito: Vide *tabela de decisão*.

Tabela de estados: Tabela que mostra as transições resultantes (válidas e inválidas) da combinação de cada estado com cada evento possível.

Taxa de falhas: Taxa do número de falhas de uma determinada categoria por uma dada unidade de medida, como, por exemplo, falhas por unidade de tempo, falhas pelo número de transações, falhas pelo número de execuções. [IEEE 610]

Técnica caixa-preta: Vide *técnica de projeto de teste caixa-preta*.

Técnica de especificação de teste: Vide *técnica de projeto de teste*.

Técnica de execução de teste: Método, manual ou automatizado, usado para executar testes.

Técnica de projeto de casos de teste: Vide *técnica de projeto de teste*

Técnica de projeto de teste: Procedimento usado para derivar ou selecionar casos de teste.

Técnica de projeto de teste baseada na especificação: Vide *técnica de projeto de teste caixa-preta*.

Técnica de projeto de teste baseada na estrutura: Vide *técnica de projeto de teste caixa-branca*

Técnica de projeto de teste caixa-branca: Procedimento para derivar e/ou selecionar casos de teste com base na análise da estrutura interna do sistema.

Técnica de projeto de teste caixa-preta: Procedimento usado para derivar e/ou selecionar casos de testes baseados na análise da especificação, funcional ou não-funcional, de um sistema sem levar em consideração sua estrutura interna.

Técnica de projeto de teste estrutural: Vide *Técnica de projeto de teste caixa-branca*

Técnica de projeto de teste funcional: Procedimento para derivar e/ou selecionar casos de teste com base na análise da especificação da funcionalidade de um sistema sem levar em consideração sua estrutura interna. Veja também *técnica de projeto de teste caixa-preta.*

Técnicas de projeto de teste não-funcional: Procedimento para derivar e/ou selecionar casos de teste para testes não-funcionais com base na análise da especificação do sistema sem levar em consideração

sua estrutura interna. Veja também *técnica de projeto de teste caixa-preta*.

Técnica de teste: Implementação da estratégia de testes para um projeto específico. A técnica normalmente inclui as decisões tomadas com base nas metas do projeto (de teste) e na avaliação de risco realizada, os pontos de partida considerando o processo de teste, as técnicas de projeto de teste que serão utilizadas, os critérios de saída e os tipos de teste que serão executados. Vide *técnica de projeto de teste*.

Técnica de teste baseada em experiência: Procedimento para derivar e/ou selecionar casos de teste com base na experiência, conhecimento e intuição do testador.

Test bed: Vide *ambiente de teste*.

Testabilidade: A capacidade que o sistema tem de permitir que as alterações feitas no software sejam testadas. [ISO 9126] Veja também *manutenibilidade*.

Testador: Um profissional capacitado que testa um sistema.

Teste: (1) Processo que contém as atividades (estáticas e dinâmicas) do ciclo de vida. Trata do planejamento, preparação e avaliação do sistema, e produtos relacionados ao trabalho para verificar se essas atividades satisfazem aos requisitos especificados, para demonstrar que estão de acordo com o objetivo e para detectar defeitos. (2) Um conjunto de ou mais casos de testes. [IEEE 829]

Teste *ad hoc*: Teste realizado informalmente; não ocorre preparação formal, não são usadas técnicas reconhecidas de projeto de teste, não há expectativa para resultados e a arbitrariedade guia a atividade da execução do teste.

Teste ágil: Prática de teste para projetos que usam metodologias ágeis, como por exemplo, o *extreme programming* (XP), que trata o desenvolvimento como cliente do teste e enfatiza o paradigma de testar antes. Veja também *desenvolvimento dirigido pelo teste*.

Teste aleatório: Técnica de projeto de teste caixa-preta onde os casos de teste são selecionados de acordo com um perfil operacional, normalmente usando um algoritmo de geração pseudo-aleatória. Esta técnica pode ser usada para testar atributos não funcionais como confiabilidade e performance.

Teste alfa: Teste operacional, simulado ou real, executado por usuários/clientes em potencial ou por uma equipe independente de teste (de fora da organização) nas instalações do desenvolvedor. O teste alfa normalmente é usado como teste de aceite interno para softwares de prateleira.

Teste *back-to-back*: Testes onde duas ou mais variantes de um sistema são executadas com as mesmas entradas. As saídas são comparadas e analisadas caso haja discrepância. [IEEE 610]

Teste baseado em código: Vide *teste caixa-branca*.

Teste baseado em projetos: Técnica de teste onde os casos de teste são projetados com base na arquitetura e/ou projeto detalhado do sistema (por exemplo: testes de interface entre componentes ou sistemas)

Teste baseado em requisitos: Técnica de teste onde os casos de teste são definidos com base nos objetivos do teste e as condições de teste derivadas dos requisitos. Por exemplo, testes que exercitam funções específicas ou investigam atributos não funcionais como confiabilidade ou usabilidade.

Teste baseado em riscos: Testes orientados para explorar e fornecer informação a respeito dos riscos do produto. [Após Gerrard]

Teste baseado na especificação: Vide *teste caixa-preta.*

Teste baseado no processo do negócio: Técnica de teste onde os casos de teste são projetados com base nas descrições e/ou conhecimento do processo do negócio.

Teste beta: Teste operacional realizado por usuários/clientes em potencial, fora das instalações do desenvolvedor, para determinar se um sistema satisfaz ou não suas necessidades e se encaixa nos processos do negócio. O teste beta normalmente é usado como um teste de aceite externo para *softwares* de prateleira para obter *feedback* do mercado.

Teste big-bang: Um tipo de teste de integração onde os elementos do software ou hardware (ou ambos) são combinados todos de uma vez em um sistema, ao invés de serem combinados em etapas. [Após IEEE 610] Veja também *teste de integração*.

Teste *bottom-up*: Um tipo de teste de integração incremental onde os componentes do nível mais baixo são testados e integrados primeiro. Este processo é repetido até que o componente do nível mais alto na hierarquia esteja testado. Facilita os testes dos componentes de nível mais alto. Veja também *teste de integração*.

Teste caixa-branca: Teste baseado na análise da estrutura interna do sistema.

Teste caixa-preta: Teste, funcional ou não-funcional, que não leva em consideração a estrutura interna do sistema.

Teste comparativo: (1) Um padrão com o qual as medidas ou comparações podem ser feitas. (2) Um teste que pode ser usado para comparar componentes ou sistemas entre si ou com um padrão como em (1). [Após IEEE 610]

Teste completo: Vide *teste exaustivo.*

Teste de aceite: Teste formal das necessidades, exigências e processos de negócio do usuário conduzidos para determinar se um sistema satisfaz ou não os critérios de aceite e para permitir que o usuário, cliente ou outra entidade autorizada determinem se aceitam ou não o sistema. [Após IEEE 610]

Teste de aceite do usuário: Vide *teste de aceite*

Teste de aceite local: Teste de aceite executado por usuários/clientes nas suas instalações, para determinar se o sistema satisfaz ou não as suas necessidades e se encaixa nos processos de negócio. Normalmente inclui hardware e software.

Teste de acessibilidade: Teste para determinar qual o grau de facilidade com que usuários com necessidades especiais conseguem usar o sistema em teste. [Gerrard]

Teste de algoritmos [TMap]: Vide *teste de ramos*.

Teste de arcos: Vide *teste de ramos*.

Teste de armazenamento: Vide *teste de utilização de recursos*.

Teste de caminhos: Técnica de projeto de teste caixa-branca onde os casos de testes são projetados para executar caminhos no software.

Teste de campos: Vide *teste beta*

Teste de capacidade de recuperação: Processo de teste para determinar a capacidade de recuperação de um sistema. Veja também *teste de confiabilidade*.

Teste de capacidade de serviço de manutenção: Vide *teste de manutenibilidade*.

Teste de carga: Teste que mede o comportamento de um sistema com uma carga crescente, como por exemplo, número de usuários paralelos e/ou números de transações, para determinar qual a carga que ele pode suportar. Veja também *teste de stress*.

Teste de casos de uso: Técnica de teste caixa-preta onde os testes são projetados para executar cenários.

Teste de cenário: Vide *teste de casos de uso*.

Teste de cobertura da lógica: Vide *teste caixa-branca*. [Myers]

Teste de comandos: Técnica de projeto de teste caixa-branca onde os casos de teste são projetados para executar comandos.

Teste de combinações de condições: Vide *teste de condições múltiplas*

Teste de combinações de condições dos ramos: Vide *teste de condições múltiplas*.

Teste de comparação elementar: Técnica de projeto de teste caixa-preta onde os casos de teste são projetados para executar combinações de entradas usando o conceito da cobertura de determinação de condições. [TMap]

Teste de compatibilidade: Vide *teste de interoperabilidade*.

Teste de complacência: Processo de teste para determinar a complacência de um sistema.

Teste de componente: Teste dos componentes de software individuais. [Após IEEE 610]

Teste de concorrência: Teste para determinar como a ocorrência de duas ou mais atividades, no mesmo intervalo de tempo, iniciadas por intercalação de atividades ou execuções simultâneas, são tratadas pelo sistema. [Após IEEE 610]

Teste de condições: Técnica de projeto de teste caixa-branca onde os casos de teste são projetados para executar as conseqüências das condições.

Teste de condições de decisão: Técnica de projeto de teste caixa-branca onde os casos de teste são projetados para executar as conseqüências das condições e das decisões.

Teste de condições múltiplas: Uma técnica de projeto de teste caixa-branca onde os casos de teste são projetados para executar combinações das conseqüências de cada condição (dentro de um comando)

Teste de confiabilidade: Processo de teste para determinar a confiabilidade de um sistema.

Teste de confiança: Vide *teste de fumaça.*

Teste de configuração: Vide *teste de portabilidade*.

Teste de confirmação: Vide *reteste.*

Teste de conformidade: Vide *teste de complacência*

Teste de conversão: Teste usado para converter dados de um sistema existente para serem usados em sistemas que o substituam.

Teste de decisão: Testes que executam as conseqüências das decisões.

Teste de desempenho: Teste que determina o desempenho de um sistema. Veja também *teste de eficiência*.

Teste de desenvolvimento: Teste formal ou informal conduzido durante a implementação do sistema. Geralmente é executado pelos desenvolvedores no ambiente de desenvolvimento. [Após IEEE 610]

Teste de determinação de condições: Teste que executa uma condição cuja conseqüência afeta independentemente a conseqüência de uma decisão.

Teste de documentação: Testa a qualidade dos documentos como, por exemplo, guia do usuário ou de instalação.

Teste de eficiência: Teste que determina a eficiência do produto do software.

Teste de entrada: Uma instância do teste de sanidade que decide se o sistema está pronto para testes mais detalhados. Um teste de entrada normalmente é feito no início da fase da execução do teste. Veja também *teste de fumaça*.

Teste de escalabilidade: Teste que determina a escalabilidade de um sistema.

Teste de estados: Vide *teste de transição de estados*

Teste de fluxo de dados: Teste que executa definições e usa pares de variáveis.

Teste de fumaça: Subconjunto de todos os casos de teste definidos/planejados que cobrem as principais funcionalidades do sistema, para verificar se eles funcionam, mas sem se preocupar com detalhes. Uma construção diária e o teste de sanidade estão entre as boas práticas da indústria. Veja também *teste de entrada*.

Teste de funcionalidade: Teste que verifica a funcionalidade de um sistema.

Teste de grafos causa-efeito: Uma técnica de projeto de teste caixa-preta onde os casos de teste são definidos a partir de grafos causa-efeito. [BS 7925/2]

Teste de instalabilidade: Testa a instalabilidade de um sistema. Veja também *teste de portabilidade*.

Teste de integração: Teste executado para encontrar defeitos nas interfaces e nas interações entre componentes ou sistemas integrados. Veja também *teste de integração de componentes, teste de integração de sistemas.*

Teste de integração de componentes: Testes executados para revelar defeitos nas interfaces e interações entre componentes integrados.

Teste de integração de sistemas: Testa a integração dos sistemas e pacotes; testa as interfaces com organizações externas (por exemplo, troca eletrônica de dados, Internet).

Teste de integração no grande: Vide *teste de integração de sistemas*.

Teste de integração no pequeno: Vide *teste de integração de componentes.*

Teste de integridade do banco de dados: Testa os métodos, processos e regras usados para acessar e gerenciar os (bancos de) dados, para garantir que eles funcionem como esperado, e que durante o acesso ao banco de dados, os dados não são corrompidos, apagados, alterados ou criados de forma inesperada.

Teste de integridade dos dados: Vide *teste de integridade do banco de dados*.

Teste de interface: Um tipo do teste de integração que trata do teste das interfaces entre os componentes ou sistemas.

Teste de interoperabilidade: Teste que determina a interoperabilidade do sistema. Veja também *teste de funcionalidade*.

Teste de LCSAJ: Uma técnica de projeto de teste caixa-branca onde os casos de teste são projetados para executar os LCSAJs.

Teste de ligação: Vide *teste de integração de componentes*.

Teste de manutenção: Teste das alterações em um sistema ou dos impactos de um ambiente modificado em um sistema.

Teste de manutenibilidade: Teste que determina a manutenibilidade de um sistema.

Teste de mesa: Teste que simula manualmente a execução de um software ou especificação. Veja também *análise estática*.

Teste de migração: Vide *teste de conversão*.

Teste de módulos: Vide *teste de componente*.

Teste de mutação: vide *teste back-to-back.*

Teste de padrão: Vide *teste de complacência*.

Teste de partição: Vide *particionamento por equivalência*. [Beizer]

Teste de portabilidade: Teste que determina a portabilidade de um sistema.

Teste de ramos: Teste que executa os ramos do software.

Teste de recuperação: Vide *teste de capacidade de recuperação.*

Teste de regressão: Teste de um programa previamente testado e que foi alterado, para garantir que todas as partes não modificadas foram cobertas ou que não foram introduzidos defeitos nas partes não modificadas do software. Ele deve ser executado quando o software ou o ambiente for modificado.

Teste de robustez: Teste que determina a robustez do sistema.

Teste de sanidade: Vide *teste de fumaça*.

Teste de segurança: Teste que determina o quão seguro é um sistema. Veja também *teste de funcionalidade*.

Teste de sintaxe: técnica de projeto de teste caixa-preta onde os casos de testes são projetados com base na definição dos domínios de entrada e/ou saída.

Teste de sistema: Processo de testar um sistema integrado para verificar se ele atende aos requisitos especificados. [Hetzel]

Teste de stress: Teste que avalia a execução de um sistema ou componente no limite (ou acima dele) dos requisitos especificados. [IEEE 610] Veja também *teste de carga*.

Teste de tabela de decisão: Técnica de projeto de teste caixa-preta onde os casos de teste são projetados para executar as combinações de entradas e/ou estímulos (causas) mostrados na tabela de decisão. [Veenendaal]

Teste de *threads*: Uma variação do teste de integração de componentes onde a integração progressiva segue a implementação de subconjuntos de requisitos e não níveis de hierarquia.

Teste de transição de estados: Teste que executa transições de estados válidas e inválidas. Veja também *teste N-switch*.

Teste de usabilidade: Teste que determina o quanto o sistema é entendido, fácil de aprender, fácil de ser operado e atraente para o usuário. [Após ISO 9126]

Teste de utilização de recursos: Teste que determina a utilização de recursos de um sistema. Veja também *teste de eficiência*.

Teste de valor limite: Vide *análise de valor limite.*

Teste de volume: Teste que submete o sistema a grandes volumes de dados. Veja também *teste de utilização de recursos*.

Teste dinâmico: Teste que envolve a execução de um software.

Teste dirigido pela lógica: Vide *teste caixa-branca*.

Teste dirigido por dados: Técnica de roteirização que armazena os dados de entrada e os resultados esperados dos testes em uma tabela ou planilha, assim, um script único de controle pode executar todos os testes da tabela. Esse teste normalmente é usado para apoiar o uso de ferramentas de execução de teste tais como ferramentas de captura/reprodução dos testes. [Fewster e Graham] Veja também *teste dirigido por palavra chave.*

Teste dirigido por palavra chave: Técnica que usa arquivos de dados para armazenar os dados de teste, os resultados esperados e as palavras chave relacionadas à aplicação que está sendo testada. As palavras chave são interpretadas por *scripts* especiais de apoio que são chamados pelos *scripts* de controle para o teste. Veja também *teste dirigido por dados*.

Teste do ciclo do processo: Teste que executa procedimentos e processos de negócio. [TMap]

Teste do perfil operacional: Teste estatístico que usa um modelo de operações do sistema (atividade de curta duração) e sua probabilidade de uso. [Musa].

Teste do programa: Vide *teste de componente.*

Teste do regulamento: Vide *teste de complacência*.

Teste do usuário: Teste onde os usuários reais avaliam a usabilidade do sistema.

Teste estático: Teste de um sistema na fase de especificação ou implementação sem executar o software (exemplo, revisão ou análise estática do código).

Teste estatístico: Teste que usa um modelo de distribuição estatística das entradas para construir casos de testes representativos. Veja também *teste do perfil operacional.*

Teste estrutural: Vide *teste caixa-branca*.

Teste exaustivo: Teste que inclui todas as combinações possíveis de valores de entrada e pré-condições.

Teste exploratório: Teste informal onde o testador controla ativamente os testes e como eles serão executados. Ele usa a informação obtida enquanto testa para projetar testes novos e melhores. [Após Bach]

Teste funcional: Teste baseado na análise da especificação da funcionalidade do sistema. Veja também *teste caixa-preta*.

Teste glass-box: Vide *teste caixa-branca*.

Teste incremental: Teste onde os sistemas são integrados e testados um (ou alguns) de cada vez, até que todos os sistemas tenham sido integrados e testados.

Teste inválido: Teste que usa valores de entrada que devem ser rejeitados pelo sistema. Veja também *tolerância a erros*.

Teste isolado: Teste dos componentes isolados dos componentes vizinhos. Esses útimos são simulados por *drivers* e *stubs*, se necessário.

Teste não-funcional: Teste dos atributos de qualidade (e não de funcionalidade) de um sistema tais como confiabilidade, eficiência, usabilidade, manutenibilidade e portabilidade.

Teste negativo: Seu objetivo é mostrar que o sistema não funciona. O teste negativo está mais relacionado à atitude dos testadores do que a alguma técnica de teste ou de projeto de teste específica. Por exemplo, testar com valores de entrada inválidos ou exceções. [Após Beizer]

Teste N-switch: Um tipo de teste de transição de estados onde o teste executa todas as seqüências válidas das N+1 transições. [Chow] Veja também *teste de transição de estados*.

Teste operacional: Teste que avalia um sistema em seu ambiente operacional. [IEEE 610]

Teste aos pares: Duas pessoas, por exemplo, dois testadores, um desenvolvedor e um testador ou um usuário final e um testador, trabalhando juntas para encontrar defeitos. Normalmente, eles usam o mesmo computador e negociam o controle do mesmo enquanto testam.

Teste "sujo": Vide *teste negativo.*

Teste *top-down*: Um tipo de teste de integração incremental onde o componente do alto da hierarquia é testado primeiro, com os componentes do nível mais baixo sendo simulados por *stubs*. Os componentes testados são usados testar os componentes dos níveis mais baixos. O processo é repetido até que os componentes do nível o mais baixo estejam testados. Veja também *teste de integração*.

Teste unitário: Vide *teste de componente*.

Testware: Artefatos, produzidos durante o processo do teste, que são necessários para planejar, projetar e para executar testes, tais como documentação, *scripts* de teste, entradas, resultados esperados, procedimentos de instalação e limpeza, arquivos, bases de dados, ambientes e qualquer outro software ou utilidades adicionais usados no teste. [Após Fewster Graham].

Testware automatizado: *Testware* utilizado nos testes automatizados, como por exemplo, *scripts* para ferramentas.

Tipo do teste: Conjunto de atividades de teste com o propósito de testar um sistema de acordo com os objetivos de um teste específico (por exemplo, teste funcional, de usabilidade, de regressão, etc). O mesmo tipo do teste pode ser executado em um ou mais níveis de teste. [Após TMap]

Tolerância à falhas: A capacidade que o sistema tem de manter o nível de desempenho especificado em caso de falhas (defeitos) no software ou de violação da interface especificada. [ISO 9126] Veja também *confiabilidade*.

Tolerância a erros: A habilidade do sistema ou componente em continuar operando normalmente apesar da presença de entradas errôneas. [Após IEEE 610]

Transição de estados: transição entre dois estados de um sistema.

Tratamento de exceção: Comportamento de um sistema em resposta a entradas errôneas vindas de um usuário, outro sistema ou de falhas internas.

Trilha de auditoria: Um caminho pelo qual a entrada original do processo (por exemplo, dados) pode ser rastreada tomando a saída do processo como ponto de partida. Isto facilita a análise do defeito e permite que seja realizada uma auditoria no processo. [Após TMap]

Tutorial de instalação: Instruções, fornecidas em mídia apropriada, que conduzem o processo de instalação. Normalmente executa o processo de instalação, fornece *feedback* sobre os resultados da instalação e mostra as opções existentes.

U

Unidade: Vide *componente*

Usabilidade: A capacidade que o software tem de ser entendido, aprendido, usado e atraente para o usuário. [ISO 9126].

Utilização de recursos: A capacidade que um sistema tem de utilizar quantidade e tipos de recursos apropriados (por exemplo, a quantidade de memória primária e secundária usada por um programa e o tamanho de arquivos temporários e de *overflow*), quando o sistema executa suas funções. [Após ISO 9126] Veja também *eficiência*.

V

Validação: Confirmação através de exame e do fornecimento de evidências objetivas que os requisitos para uso ou aplicação específicos são atendidos. [ISO 9000]

Valor de entrada: Instância de uma entrada. Veja também *entrada.*

Valor de saída: Instância de uma saída. Veja também *saída*.

Valor limite: Um valor de entrada ou saída que está nos limites de uma classe de equivalência ou na menor distância incremental do limite, tanto para baixo quanto para cima, como por exemplo, o valor mínimo ou máximo de uma escala.

Variável: Elemento de armazenamento em um computador que é acessível através de um programa e referenciado por um nome.

Verificação: Confirmação através de exame e do fornecimento de evidências objetivas que os requisitos especificados foram atendidos [ISO 9000]

Verificador: Vide *revisor*.

W

Walkthrough: Apresentação passo a passo feita pelo autor de um documento para reunir informações e estabelecer uma compreensão comum do seu conteúdo. [Freedman and Weinberg, IEEE 1028] Veja também *revisão por pares.*

Walkthrough estruturado: Vide *walkthrough.*

-x-

Anexo A (Informativo)

Índice de fontes; as seguintes fontes não nomativas foram utilizadas para construir este glossário:

[Abbott] J. Abbot (1986), Software Testing Techniques, NCC Publications.

[Adrion] W. Adrion, M. Branstad and J. Cherniabsky (1982), Validation, Verification and
Testing of Computer Software, in: Computing Surveys, Vol. 14, Nº 2, June 1982.

[Bach] J. Bach (2004), Exploratory Testing, in: E. van Veenendaal, The Testing Practitioner – 2nd edition, UTN Publishing, ISBN 90-72194-65-9.

[Beizer] B. Beizer (1990), Software Testing Techniques, van Nostrand Reinhold, ISBN 0-442- 20672-0

[Chow] T. Chow (1978), Testing Software Design Modelled by Finite-Sate Machines, in:
IEEE Transactions on Software Engineering, Vol. 4, Nº 3, May 1978.

[CMM] M. Paulk, C. Weber, B. Curtis and M.B. Chrissis (1995), The Capability Maturity
Model, Guidelines for Improving the Software Process, Addison-Wesley, ISBN 0-201-54664-7

[CMMI] M.B. Chrissis, M. Konrad and S. Shrum (2004), CMMI, Guidelines for Process Integration and Product Improvement, Addison Wesley, ISBN 0-321-15496-7

[Fenton] N. Fenton (1991), Software Metrics: a Rigorous Approach, Chapman & Hall, ISBN0-53249-425-1

[Fewster and Graham] M. Fewster and D. Graham (1999), Software Test Automation,
Effective use of test execution tools, Addison-Wesley, ISBN 0-201-33140-3.

[Freedman and Weinberg] D. Freedman and G. Weinberg (1990), Walkthroughs, Inspections, and Technical Reviews, Dorset House Publishing, ISBN 0-932633-19-6.

[Gerrard] P. Gerrard and N. Thompson (2002), Risk-Based E-Business Testing, Artech House Publishers, ISBN 1-58053-314-0.

[Gilb and Graham] T. Gilb and D. Graham (1993), Software Inspection, Addison-Wesley,
ISBN 0-201-63181-4.

[Grochtmann] M. Grochtmann (1994), Test Case Design Using Classification Trees, in:
Conference Proceedings STAR 1994.

[Hetzel] W. Hetzel (1988), The complete guide to software testing – 2nd edition, QED
Information Sciences, ISBN 0-89435-242-3.

[McCabe] T. McCabe (1976), A complexity measure, in: IEEE Transactions on Software
Engineering, Vol. 2, pp. 308-320.

[Musa] J. Musa (1998), Software Reliability Engineering Testing, McGraw-Hill Education,
ISBN 0-07913-271-5.

[Myers] G. Myers (1979), The Art of Software Testing, Wiley, ISBN 0-471-04328-1.

[TMap] M. Pol, R. Teunissen, E. van Veenendaal (2002), Software Testing, A guide to the
TMap Approach, Addison Wesley, ISBN 0-201-745712.

[Veenendaal] E. van Veenendaal (2004), The Testing Practitioner – 2nd edition, UTN
Publishing, ISBN 90-72194-65-9.

Anexo B (Método para comentar este glossário)

Agradecemos os comentários e sugestões enviadas para que o glossário possa ser melhorado satisfazendo as necessidades da comunidade de testes.

Para sugestões de melhorias e inclusão de novos termos, favor contactar:
nannini@ibqts.com.br

Quando enviar um comentário, garanta as seguintes informações:
- Seu nome e contato;
- A versão do glossário;
- Parte exata do glossário;
- Razão da proposta de mudança ou referência do termo e definição. Os créditos serão dados na próxima versão do glossário.