TESTES DE SOFTWARE E GERÊNCIA DE CONFIGURAÇÃO

Jeanine dos Santos Barreto

SOLUÇÕES EDUCACIONAIS INTEGRADAS

Tipos de teste

Objetivos de aprendizagem

Ao final deste texto, você deve apresentar os seguintes aprendizados:

- Explicar os fundamentos dos testes de software.
- Descrever a verificação, a validação estática, a validação dinâmica e as categorias de teste de software.
- Diferenciar os tipos de teste de software e suas etapas (o que testar, como testar e quando testar).

Introdução

Existem vários tipos de testes de software. Eles podem ser realizados por diferentes pessoas, em diversas fases do projeto e com objetivos distintos. É importante conhecê-los para entender a sua importância e a sua aplicação durante todo o ciclo de vida do projeto de software.

Neste capítulo, você vai estudar os fundamentos e os tipos de testes de software. Também vai conhecer a verificação, a validação estática, a validação dinâmica e as categorias de testes de software. Além disso, vai ver os tipos de teste de software e as suas etapas.

Fundamentos dos testes de software

As áreas de engenharia costumam combinar atividades de projeto e construção com atividades de verificação intermediária do que está sendo produzido e de verificação final do produto pronto. Tudo isso seve para que as falhas, erros e defeitos sejam identificados, reduzidos e até mesmo eliminados por completo.

Na ciência da computação, a disciplina de engenharia de software trata seus produtos informatizados da mesma forma. Durante o trabalho de desenvolvimento de um software de qualidade, tanto as atividades de projeto quanto as atividades de verificação têm papel fundamental para o sucesso final. As atividades de verificação podem ser entendidas como as atividades de testes.

Como todo projeto, o desenvolvimento de um produto de software envolve uma equipe formada por pessoas. Naturalmente, durante o ciclo de vida do projeto, é necessária a comunicação entre os integrantes da equipe. É principalmente por esse motivo que os problemas nos softwares acontecem. E também é por isso que as atividades de teste são tão importantes. Afinal, as pessoas possuem formações, lógicas, opiniões e maneiras de trabalhar diferentes. Mesmo que todas trabalhem juntas para atingir o mesmo objetivo, tendem a realizar suas atividades da maneira mais fácil, o que nem sempre pode resultar em um trabalho coeso e funcional ao final do projeto.

As atividades de teste são fundamentais para os projetos de desenvolvimento de software, principalmente pelos seguintes motivos (PEZZÈ; YOUNG, 2008):

- descobrir defeitos, falhas e erros no software antes que ele seja entregue ao cliente;
- demonstrar que o software atende aos requisitos e às necessidades dos usuários que vão realizar as suas tarefas por meio dele;
- garantir maior segurança aos usuários na utilização do software entregue;
- aumentar a possibilidade de continuidade do negócio dos clientes;
- melhorar a qualidade dos softwares produzidos, devido à experiência adquirida com os projetos desenvolvidos;
- melhorar a imagem do software e da própria fábrica de software perante os clientes e usuários, por fornecer produtos que atendem às suas expectativas;
- diminuir ao máximo o tempo e as despesas gastos na correção de problemas.

As atividades de projeto e de testes podem ser utilizadas para projetos de diferentes tipos. Isso inclui desde aqueles mais simples até os mais complexos, que resultam em produtos extremamente customizados pelo cliente e cujo sucesso é altamente necessário. A escolha dessas atividades depende do tipo de projeto, do produto que se quer produzir, dos requisitos de qualidade e da maneira como será feito o desenvolvimento.

As atividades de testes dos produtos fabricados em série envolvem um conjunto predefinido e sistematizado de análises e testes. Tais análises e testes são capazes de indicar se o produto satisfaz o nível de qualidade esperado para ele. Os softwares, por sua vez, por serem mais personalizados de acordo com o cliente, precisam de conjuntos específicos de análises e testes. A ideia é atender também a níveis de qualidade individuais de cada projeto. Quanto mais complexo for o software, mais complexas serão as atividades de verificação e testes envolvidas.

Fique atento

Nas indústrias, onde a fabricação de produtos é muito repetitiva, é comum que as atividades de verificação sejam feitas em apenas partes dos produtos ou nos produtos finais, mas em esquema de amostragem. Ou seja, apenas algumas peças são escolhidas para serem testadas. Isso acontece porque, devido ao alto nível de padronização, o processo de fabricação já é bastante confiável. Assim, testar cada produto de maneira individual poderia trazer uma despesa extra desnecessária.

Por outro lado, se os produtos forem computadores, carros, aviões, navios, instrumentos cirúrgicos, cofres e outros de grande importância, mesmo que sejam fabricados em série, é necessário verificá-los de maneira individual antes que sejam entregues aos seus usuários finais. Isso acontece porque são artefatos em que a qualidade precisa ser muito elevada e as falhas nem sempre são toleradas.

Já no caso dos softwares, os testes e as atividades de verificação são essenciais e devem ser feitos sempre de maneira individualizada. O motivo é que, via de regra, cada software produzido atende às expectativas de um só cliente. Nesse sentido, o software deve ser testado durante o seu desenvolvimento, até a entrega ao usuário final.

O custo das atividades de análise, verificação e testes geralmente corresponde a mais da metade do custo total previsto para todo o desenvolvimento e ainda para a manutenção de um projeto de software. Já existem algumas ferramentas e técnicas que podem ser utilizadas em tempo de programação para diminuir os erros e falhas de desenvolvimento. Contudo, ainda não existe uma maneira de eliminar completamente os problemas de um software para que ele seja entregue perfeito ao usuário final.

Apesar de parecer um paradoxo, a verdade é que a tentativa de destruir a confiabilidade de um software por meio da identificação de erros, defeitos e falhas resulta no aumento da sua confiabilidade. Isso acontece pois, uma vez corrigidos todos os problemas, o produto entregue ao cliente atende objetivamente às suas necessidades, tornando agradável a experiência de utilizá-lo.

É importante você notar que, quanto mais cedo os testes forem iniciados, menores serão o custo e o prazo utilizados na correção dos problemas encontrados. Por isso, existem estágios pelos quais os testes de software normalmente passam (PRESSMAN, 2011). Veja a descrição dos estágios a seguir e depois observe a Figura 1.

- **Planejamento:** elaboração e revisão da estratégia e do plano de teste.
- **Preparação:** preparação do ambiente de testes, incluindo equipamentos, softwares, rede, pessoal e ferramentas.

- **Procedimentos iniciais:** elaboração de um documento que estabelece o acordo entre as partes envolvidas nos testes (objetivo, pessoal, responsabilidades, etc.).
- **Especificação:** elaboração e revisão dos casos de teste manuais (roteiros de testes) e automáticos (*scripts*).
- **Execução:** execução dos testes planejados conforme os casos de teste elaborados e geração dos registros de resultados (relatórios de testes).
- **Entrega:** conclusão do processo de testes com a entrega da parte ou do todo do sistema.

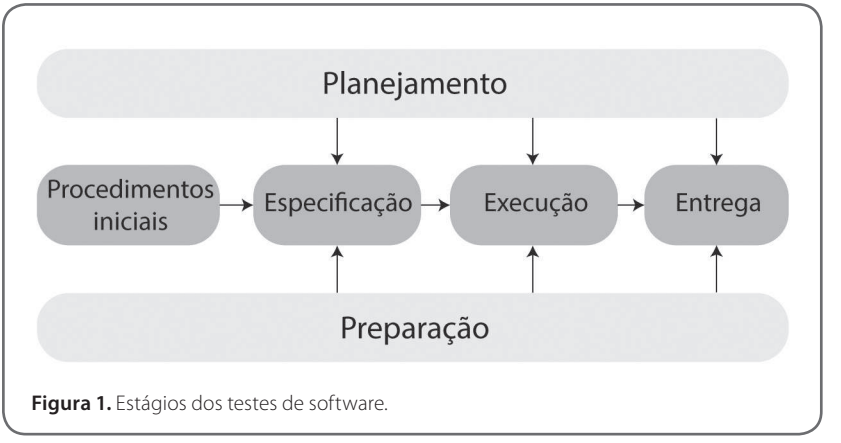

Figura 1. Estágios dos testes de software.

Os problemas que um software pode apresentar são inúmeros e diferenciados para cada projeto, assim como as maneiras de identificá-los e de corrigi-los. É por esse motivo que não existe uma maneira correta e predefinida de estabelecer as atividades e etapas envolvidas nos testes de software. É preciso analisar o projeto em questão para identificar a melhor maneira de aplicar as atividades de testes ao longo do desenvolvimento do produto de software.

Verificação, validação estática, validação dinâmica e categorias de teste de software

Antigamente, era comum que a fabricação dos softwares englobasse todas as atividades de análise e testes apenas no final do ciclo de vida de desenvolvimento. Existia um profissional que se chamava testador, e a sua atividade básica consistia em realizar a execução de casos de teste para um produto completamente pronto.

Atualmente, as empresas fabricantes de softwares já conseguem compreender melhor que a realização dos testes faz parte do processo de verificação e validação. Tal processo, por sua vez, é fundamental para a avaliação e para a manutenção da qualidade requerida de um produto de software.

A verificação e a validação têm como propósito principal garantir que o software esteja adequado aos seus objetivos e que atenda às expectativas dos clientes. Em outras palavras, essas etapas garantem que o software cumpra as suas especificações, aquilo a que ele se propõe (PEZZÈ; YOUNG, 2008).

Na **verificação**, existe uma análise para entender se o software atende aos requisitos funcionais e não funcionais que foram definidos para ele. A atividade de **validação** consiste em verificar se o software atende às expectativas e necessidades do usuário final. Apesar de esses dois termos terem significados diferentes, as duas atividades não acontecem de maneira independente ou sequencial.

As atividades de verificação e de validação têm início até mesmo antes que um projeto de desenvolvimento de software seja idealizado. Isso porque é preciso que os gestores de projeto e de tecnologia de informação tenham em mente se a sua equipe possui as competências necessárias. Ou seja, eles devem verificar se a sua equipe tem o conhecimento, a habilidade e a atitude necessárias para desenvolver produtos de software que satisfaçam às necessidades e expectativas dos usuários. Essa verificação se chama **estudo de viabilidade**.

Fique atento

Na área da administração, a competência de um funcionário é definida pelo conjunto de três elementos. Veja a seguir.

- **Conhecimento:** é o saber. É tudo o que se adquire por meio de cursos, treinamentos, livros, experiência de vida.
- **Habilidade:** é o saber fazer. É aquilo que depende de prática, de treino, de saber lidar com os acertos e com os erros cometidos.
- **Atitude:** é o querer fazer. É a ação necessária para colocar em prática os conhecimentos e a habilidade. Normalmente, é o elemento essencial para mudanças de procedimentos.

Se o estudo de viabilidade for positivo e indicar que a equipe é capacitada para participar do projeto, então as atividades de verificação e validação começam. Elas ocorrem juntamente às atividades de desenvolvimento, tendo fim depois que o produto final for liberado para o cliente.

Atualmente, é natural que os produtos de software sejam entregues em etapas, passando por reavaliações e ajustes dos requisitos, caso sejam necessários. Por esse motivo, é muito importante que as atividades de verificação e validação sejam pensadas de antemão, para que se consiga cumprir a previsão de custos e prazos.

O estudo de viabilidade é a primeira atividade que deve ser realizada em um projeto de software rumo à entrega de um produto satisfatório. Esse estudo está aliado às atividades de verificação, validação e desenvolvimento. São as atividades de verificação que avaliam se o produto que está sendo construído tem coerência com os requisitos e com a qualidade esperada, tanto nas entregas em etapas como no produto final. No caso das atividades de validação, é avaliado se existe coerência entre os produtos intermediários e o final apresentados ao usuário. Além disso, são avaliadas as expectativas do usuário para o software (PEZZÈ; YOUNG, 2008).

Em seguida, é preciso escolher o conjunto de técnicas de teste e de análise necessárias. Para isso, é preciso levar em consideração o nível de qualidade esperado, o custo, o prazo e os recursos disponíveis para o desenvolvimento do software. É possível combinar as técnicas e as ferramentas disponíveis, o que pode trazer ganhos para o andamento do projeto. Em síntese, nenhuma técnica de teste ou análise serve sozinha a todos os objetivos. A seguir, você pode ver as razões primárias para combinar técnicas (PEZZÈ; YOUNG, 2008).

- **Eficácia para diferentes classes de erros:** por exemplo, condições de corrida são muito difíceis de se encontrar com teste convencional, mas podem ser detectadas com análise estática.
- **Aplicabilidade em diferentes etapas do projeto:** por exemplo, você pode aplicar técnicas de inspeção inicialmente aos requisitos e representações de projeto que não são apropriados para análises mais automáticas.
- **Diferenças de objetivos:** por exemplo, o teste sistemático (não randômico) busca maximizar a detecção de falhas, mas não pode ser usado para medir confiabilidade; para isso, o teste estatístico é necessário.
- **Compromissos entre custo e garantias:** por exemplo, você pode utilizar uma técnica relativamente custosa para garantir algumas propriedades essenciais de componentes centrais (como um *kernel* de segurança), enquanto as mesmas técnicas seriam caras demais se aplicadas a todo o projeto.

Logo no início do desenvolvimento do projeto, o detalhamento dos requisitos do software deve ser escrito de maneira que seja possível fazer a verificação automatizada ou de forma manual, pelos próprios desenvolvedores e testadores. A lista de verificação que deve ser feita pode ser fundamentada em experiências com erros anteriores, de outros projetos, ou somente nos requisitos identificados para cada parte do software.

É importante você entender que, quando a validação é feita utilizando padrões e ferramentas automatizadas de testes, esse tipo de validação é chamado de **estático**. Já quando a validação é feita de maneira diferente para cada funcionalidade ou para cada sistema, levando-se em conta que cada produto de software tem suas características próprias, esse tipo de validação se chama **dinâmico** e normalmente é feito de maneira manual.

Mais adiante, o plano de teste precisa se basear nas especificações de requisitos do software, no código-fonte dos programas e em toda a documentação disponível. Via de regra, as inspeções de código-fonte se resumem em solicitar que seja feita uma revisão por outro desenvolvedor que não aquele que programou o código. Isso porque é mais provável que alguém que não realizou a tarefa consiga identificar problemas e erros.

Durante o desenvolvimento, os desenvolvedores realizam testes unitários de funcionalidades para verificar se cada parte do software, isoladamente, realiza as atividades que foram solicitadas. Nesse ponto, é preciso que sejam documentadas todas as diferenças entre aquilo que era esperado e o que é produzido até o momento.

Quando o projeto já está mais adiantado, os casos de teste se apresentam como uma técnica para, principalmente, verificar as interfaces com os usuários e identificar o quanto os testes unitários utilizam cada interface. Isso serve para indicar se as especificações de interface estão completas ou não.

Por fim, os testes de integração, que são elaborados pela equipe de qualidade ou por aqueles analistas que tiverem esse papel, testam as funcionalidades de maneira integrada. A ideia é indicar se elas funcionam quando precisam se comportar como um sistema. São testados os diferentes caminhos possíveis dentro dos programas, identificando se o sistema funciona de maneira global.

Testes de software e suas etapas

Ainda que o planejamento do desenvolvimento de um produto de software tenha sido elaborado de maneira detalhada e cuidadosa, é praticamente impossível que as partes do software ou o produto final sejam entregues sem algum tipo de problema, como defeitos, erros e falhas (PEZZÈ; YOUNG, 2008).

Saiba mais

Um produto de software pode apresentar problemas. A seguir, veja como esses problemas podem ser classificados.

- **Erro:** é qualquer erro humano que resulta em algo incorreto. Um erro acontece quando existe uma diferença entre o valor obtido em uma operação matemática e o valor que era esperado.
- **Defeito:** é qualquer inconsistência do software, evidenciando algo que foi implementado ou programado de maneira incorreta. Um exemplo acontece quando existem incorreções nas linhas de código-fonte. Todo defeito é causa de erros, mas, por outro lado, se a linha que estiver incorreta jamais for executada, o erro não vai acontecer.
- **Falha:** é qualquer comportamento inesperado realizado pelo software. Uma falha pode ter sido gerada por vários erros, mas alguns erros podem nunca gerar uma falha. É o caso de uma operação aritmética que é programada e não é usada.

Os testes de software servem para identificar de maneira antecipada todo e qualquer erro, defeito ou falha. Assim, é possível evitar que eles apareçam quando o software for entregue para o cliente. Dessa maneira, as correções são feitas antes da finalização. Antigamente, era normal que os testes fossem

realizados somente depois de finalizada a fase de desenvolvimento. A ideia era, dessa maneira, encontrar todos os problemas e resolvê-los.

Atualmente, é de conhecimento geral que é preciso efetuar testes ao longo de todo o desenvolvimento para evitar custos e prazos diferentes daqueles previstos inicialmente. Afinal, determinados problemas, quando encontrados, podem significar o retorno de todo um projeto para as fases iniciais.

Nesse sentido, existem vários tipos de testes que podem ser realizados durante um projeto de software. Tais testes se aplicam a diferentes propósitos, como você pode ver a seguir (PRESSMAN, 2011).

- **Teste unitário:** normalmente é realizado pelos próprios desenvolvedores, nas porções menores do software que estão sendo desenvolvidas no momento, de maneira individualizada. Ele serve para verificar se as partes do software funcionam de maneira isolada das demais partes do sistema.
- **Teste de caixa branca:** avalia a parte interna do software, seu código-fonte. Ele serve para identificar problemas na lógica de programação e também na estrutura do programa, observando elementos como as condições usadas, os laços de repetição e o fluxo tomado pelos dados. O testador deve lembrar-se de verificar se o nível de segurança e confiança exigido está sendo implementado.
- **Teste de caixa preta:** avalia a parte externa do software, o seu modo de funcionamento. Esse tipo de teste serve para identificar se o software está funcionando como deveria, se os dados informados resultam nas informações pretendidas e se, de maneira geral, o sistema faz o que ele deveria fazer.
- **Teste de caixa cinza:** é uma combinação dos testes de caixa branca e de caixa preta, pois ele avalia os aspectos internos e também os aspectos externos do software, as suas entradas, o fluxo dos dados e as saídas.
- **Teste de regressão:** tem a função de testar cada nova versão do software toda vez que uma funcionalidade for modificada. Ou seja, toda a parte pronta do software será testada novamente para verificar se alguma novidade resultou em problema. Esse tipo de teste é muito importante para indicar se erros que já haviam sido corrigidos não voltaram a se manifestar.
- **Teste de integração:** tem como propósito verificar se as porções menores, testadas anteriormente pelos testes unitários, têm condições de funcionar em conjunto, formando um sistema. Esse teste é muito importante, pois funcionalidades que operam perfeitamente de maneira

isolada podem apresentar problemas quando tentam, por exemplo, realizar o fluxo de dados de uma parte do sistema para outra.

- **Teste de volume:** esse teste tem como objetivo avaliar até que limites um software pode ser utilizado, ou seja, qual é o seu limite de suporte a informações ou tráfego sem que apresente nenhum problema.
- **Teste de performance:** divide-se em outros três tipos, listados a seguir.
  - Teste de carga: verifica o software como um todo, em condições normais de uso, avaliando o tempo de resposta das operações, quantas operações podem ser executadas em determinado período de tempo, quantos usuários simultâneos gravando dados podem existir, entre outros aspectos.
  - Teste de estresse: identificados os limites do software por meio dos testes de volume, esse teste serve para levar o software aos seus limites. A ideia é verificar em que ponto ele para de funcionar corretamente.
  - Teste de estabilidade: verifica se o software se mantém funcionando de maneira adequada depois de ser utilizado por um longo período de tempo.
- **Teste funcional ou de funcionalidade:** verifica se o software como um todo, bem como cada parte dele, faz exatamente o que deveria fazer, ou seja, se os casos de uso foram corretamente descritos e desenvolvidos.
- **Teste de segurança:** verifica se o software permite que os dados sejam acessados somente pelos perfis determinados para cada parte específica do sistema ou para cada funcionalidade.
- **Teste de usabilidade:** é realizado por usuários e não por analistas. O seu propósito é verificar se o software satisfaz as necessidades do usuário. Esse teste serve para identificar a forma como o usuário utiliza as funcionalidades do software, na tentativa de apontar partes do sistema em que ele apresenta maior dificuldade de interação. Como esse teste é feito pelos clientes, é importante o acompanhamento de analistas para documentar comentários, sugestões e reclamações.
- **Teste de aceitação:** serve para verificar se o produto de software está pronto para ser entregue ao cliente, ou seja, se ele está pronto para entrar em produção. Geralmente, esse teste é realizado por alguém indicado pelo cliente, ou ainda por um testador especializado que não tenha participado do projeto, mas que tenha grande conhecimento acerca dos requisitos do software.

- **Teste de instalação:** tem como propósito verificar se o software é passível de ser instalado de maneira correta, em diferentes hardwares, com diferentes sistemas operacionais e diferentes disposições de memória, rede, entre outros aspectos.
- **Teste de configuração:** serve para identificar se o software funciona de maneira adequada no hardware para o qual foi planejado e no qual será instalado.
- **Teste de manutenção:** avalia se determinada mudança em algum aspecto do software resultou em falhas no seu funcionamento como um todo.

Referências

PEZZÈ, M.; YOUNG, M. *Teste e análise de software:* processos, princípios e técnicas. Porto Alegre: Bookman, 2008.

PRESSMAN, R. S. *Engenharia de software:* uma abordagem profissional. 7. ed. Porto Alegre: AMGH, 2011.

Encerra aqui o trecho do livro disponibilizado para esta Unidade de Aprendizagem. Na Biblioteca Virtual da Instituição, você encontra a obra na íntegra.

Conteúdo:

SAGAH
SOLUÇÕES EDUCACIONAIS INTEGRADAS