TESTES DE SOFTWARE E GERÊNCIA DE CONFIGURAÇÃO

Jeanine dos Santos Barreto

SAGAH
SOLUÇÕES EDUCACIONAIS INTEGRADAS

Níveis de teste

Objetivos de aprendizagem

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

- Definir os níveis de teste e o teste de sistema.
- Diferenciar os níveis de teste de unidade e de integração.
- Descrever os níveis de teste de aceitação e de regressão.

Introdução

Os testes de software não têm um momento específico do projeto de desenvolvimento de um sistema para acontecer. Eles devem ocorrer em todos os estágios do ciclo de vida do projeto. A cada estágio do projeto, são definidos níveis de teste. Cada um é responsável por testar elementos específicos do software e validar determinados aspectos do software que havia sido solicitado originalmente.

Neste capítulo, você vai estudar os níveis de teste. Além disso, vai conhecer os testes de sistema, de unidade, de integração, de aceitação e de regressão.

Os níveis de teste e o teste de sistema

Os testes de software são aplicados em diferentes estágios do projeto de desenvolvimento do software. Eles exigem diferentes níveis de esforço de trabalho tanto dos testadores como de toda a equipe do projeto. Basicamente, existem três práticas para a realização dos testes de software (DELAMARO; MALDONADO; JINO, 2016), como você pode ver a seguir.

- Analistas diferentes daqueles que realizaram a implementação e a codificação do software realizam os testes depois que uma funcionalidade é desenvolvida e antes que ela seja entregue ao usuário final. A desvantagem dessa prática é que os testes podem ser realizados quando o prazo de entrega do produto já está praticamente estourado. Isso pode

comprometer o tempo gasto com os testes e, pior ainda, com a correção dos problemas encontrados nos testes.

- Os testes do software começam a ser realizados ainda no início do projeto de desenvolvimento de software. Assim, as funcionalidades são testadas à medida que são desenvolvidas, num processo contínuo.
- Na programação extrema, o modelo de desenvolvimento é orientado às boas práticas da engenharia de software, sendo que uma delas é o teste. A XP não encara o teste como uma perda de tempo e uma das suas diretrizes básicas é a de que o teste deve ser feito constantemente, o tempo todo.

Saiba mais

A *extreme programming* (XP) — programação extrema, em português — é uma metodologia ágil de desenvolvimento de sistemas. Ela é voltada principalmente para equipes que precisam desenvolver softwares com requisitos não muito precisos, ou ainda que sofrem mudanças constantes, como aqueles que envolvem processos financeiros. Nesse tipo de metodologia, existe um acompanhamento constante da codificação. Além disso, os testes são realizados continuamente para que os ajustes sejam feitos ao longo do projeto de desenvolvimento.

A metodologia ágil — o *agile software development* (desenvolvimento ágil de software) — se refere à forma de desenvolvimento de software em que o produto é fabricado aos poucos, em pequenos pedaços e períodos, buscando minimizar os riscos do desenvolvimento. Esses pedaços são chamados de iterações. Cada iteração é considerada um pequeno projeto de software, incluindo todas as tarefas necessárias, do início à implantação, com foco nos testes e correções.

Em função do estágio ou do nível em que o processo de desenvolvimento do software se encontra, é possível definir os seguintes níveis de testes:

- teste de unidade;
- teste de integração;
- teste de sistema;
- teste de aceitação;
- teste de regressão.

Teste de sistema

No nível de teste de sistema, os analistas de teste, ou testadores, têm o propósito de executar o sistema como um todo, levando em consideração a visão do usuário final. Todas as funcionalidades do sistema são colocadas em operação.

Esse nível de teste tem a intenção de identificar problemas que diferenciem o software entregue daquele software original, idealizado pelos usuários e do qual havia sido feita a análise de requisitos (MARTINS, 2007). Em outras palavras, o teste de sistema tem o papel de averiguar o comportamento de todo o software para identificar se ele é ou se parece com aquele produto que havia sido definido no escopo do projeto pela equipe e pelos usuários. A ideia é realizar testes considerando os requisitos funcionais e não funcionais do software.

Saiba mais

Os requisitos de um software envolvem todas as necessidades, expectativas e objetivos do usuário em relação à utilização do sistema. Os **requisitos funcionais** são aqueles que especificam o que deve ser feito pelo sistema; são as atividades que precisarão ser realizadas por meio da sua utilização. Os requisitos funcionais normalmente são descritos com um verbo no infinitivo, como "agendar consulta de paciente", "emitir relatório de fechamento de caixa", "efetuar o fechamento das médias dos alunos", entre outros exemplos.

Por outro lado, os **requisitos não funcionais** do software definem a maneira como ele realiza as tarefas para as quais foi fabricado. É possível dizer que esse tipo de requisito tem relação com as exigências de qualidade, de segurança, de confiabilidade, de tempo de operação, entre outros aspectos. A facilidade de uso, a portabilidade para outro sistema operacional, a linguagem de programação a ser utilizada e os recursos consumidos de memória são exemplos de requisitos não funcionais.

Para a realização do teste de sistema, as atividades de teste precisam ocorrer em condições completamente semelhantes àquelas em que o software vai funcionar depois que tiver sido entregue ao usuário final. Essas condições envolvem o ambiente, as interfaces com outros sistemas, os aspectos tecnoló-

gicos, as massas de dados, o nível de experiência dos usuários e qualquer outro aspecto que se relacione com a maneira como o software vai ser executado quando estiver na sua realidade de negócio.

Além de testar o sistema como um todo, inserido em todas as condições e ambientes em que ele será utilizado, outra grande oportunidade trazida pelo nível de teste de sistema é a de realizar a validação da integração entre o sistema que está sendo (ou foi) desenvolvido e outros sistemas que possam ser utilizados juntamente a ele. Para a realização desse tipo de teste, em que se verifica a integração entre sistemas como um todo, deve haver uma responsabilização do gerente de projeto. Ele é que vai decidir se o teste precisa realmente ser realizado e também em qual momento, uma vez que não há necessidade de finalizar a codificação do sistema atual para que ele aconteça.

É comum que o nível do teste de sistema seja atingido e possa ser realizado somente quando o software estiver totalmente codificado e funcionando de maneira global, como um todo. Por isso, o melhor é adotar um ciclo de vida iterativo, a fim de que os testes possam ser realizados bem antes de o software estar pronto. Isso possibilita que o comportamento dos casos de uso seja testado bem antes da entrega.

Mesmo sendo realizado antes de todo o sistema estar pronto, o teste de sistema sempre vai ter como último objetivo o teste do funcionamento do sistema desde o início dos fluxos até o final, ou seja, de uma ponta à outra das funcionalidades.

Os níveis de teste de unidade e de integração

O nível de teste de unidade se relaciona com a validação de cada parte desenvolvida do software, normalmente realizada pelo próprio analista desenvolvedor. O teste de integração, por sua vez, tem relação com a integração entre partes do software já desenvolvidas, ou mesmo do software com outros sistemas interdependentes.

Teste de unidade

O teste de unidade também é conhecido como teste unitário ou teste de módulo. Nesse nível de teste, são testadas as menores partes em que um software pode ser dividido. O propósito principal desse tipo de teste é realizar a validação de chamadas de métodos, classes, funções, sub-rotinas e partes de código-

-fonte que possam ser testadas de maneira isolada. Nesse sentido, a intenção de um teste de unidade é identificar problemas em partes do software de maneira independente, sem efetuar testes que vinculem essas partes a outras (MARTINS, 2007).

Fique atento

Normalmente, o teste unitário, ou teste de unidade, acontece por meio do acesso ao código ou à estrutura interna do software e em ambiente de desenvolvimento, para não causar nenhum dano a dados e informações reais. É por isso que, necessariamente, o teste de unidade envolve o analista desenvolvedor que trabalhou diretamente na construção do código-fonte que está sendo testado.

Geralmente, quando um problema é encontrado no código ou na estrutura interna do software, ele é imediatamente corrigido, sem que seja necessário documentar de maneira formal a discrepância encontrada. Uma ligeira anotação é suficiente, apenas para que sirva de auxílio para os desenvolvedores não repetirem os erros encontrados enquanto codificam o restante do sistema.

Normalmente, o analista responsável pelos testes de unidade utiliza os seguintes documentos como auxílio na realização dos testes (DELAMARO; MALDONADO; JINO, 2016):

- documento de especificação de requisitos do software;
- documento de caso de uso;
- documento de arquitetura do software (DAS).

Saiba mais

O documento de arquitetura do software define uma visão geral a respeito da arquitetura do sistema do software. Ele tem o papel de estabelecer uma comunicação entre o arquiteto do software e os demais membros da equipe do projeto. Além disso, descreve toda e qualquer decisão significativa a respeito da arquitetura idealizada para o software.

De posse desses três documentos, pelo menos, o analista testador pode elaborar cenários de testes unitários, que devem ser testados por ele. O teste unitário é aquele que foca o esforço do testador em partes menores do software que têm a possibilidade de serem testáveis. Assim, é comum que sejam aplicados valores de entrada aos diversos fluxos de controle e de dados, para verificar se eles funcionam conforme o esperado.

É por esse motivo que são testadas funções, chamadas de métodos, entre outros, pois são partes pequenas que possibilitam validar se o resultado é o esperado após a entrada de um dado. Via de regra, todo desenvolvedor já realiza uma espécie de teste de unidade, uma vez que é preferível que o código seja testado à medida que é implementado.

Teste de integração

O nível do teste de integração tem como diferença principal do teste de unidade o fato de o seu propósito ser o de encontrar problemas gerados pela integração interna entre os componentes ou partes do software. É normal que, por meio do teste de integração, sejam encontrados problemas de transmissão de dados (DELAMARO; MALDONADO; JINO, 2016).

Exemplo

Para entender melhor o tipo de problema encontrado pelo teste de integração, considere o caso de uma função B que está aguardando receber determinado valor como resultado da execução de uma função A. O tipo de dado que B espera é um número inteiro, então, caso a função A retorne uma sequência de caracteres, esse será um erro detectado pelo teste de integração, que resultará em negativo.

O teste de integração também é o responsável por identificar grande parte dos problemas relacionados à interface do sistema. Em contrapartida, é interessante frisar que os problemas se relacionam à interface do software que está sendo desenvolvido atualmente, e não a outros sistemas que porventura façam integração com ele.

Esse tipo de teste, em que se integram o sistema atual e outros que possam ser utilizados por ele, são foco do nível de teste chamado de teste de sistema. Fica sob a responsabilidade do gerente de projeto decidir se eles serão feitos

ou não e em qual momento, uma vez que eles podem ser realizados antes mesmo que o software esteja completamente codificado.

Por meio do teste de integração, é possível garantir que os componentes que fazem parte do modelo que foi implementado funcionam de maneira adequada quando são combinados com outros componentes, de outros casos de uso. O propósito do teste de integração é formar um pacote ou grupos de pacotes que contenham modelos de implementação. Esses pacotes arrumados em grupo normalmente são de várias partes do código desenvolvido, inclusive de outros sistemas. Por isso, o teste de integração está num nível em que pode detectar problemas, como erros, falhas ou defeitos, na própria interface do software.

Para que o teste de integração seja realizado com sucesso, é importante que envolva analistas desenvolvedores e analistas de testes, ou testadores. Isso se deve à importância de cada um fazer a sua parte durante os testes, a fim de que a estrutura interna e a estrutura externa do software sejam integradas e testadas em sua totalidade. Para isso, é fundamental que os gestores da equipe do projeto de desenvolvimento de software esclareçam todos os integrantes a respeito da visão de qualidade que se deseja atingir, para que todos trabalhem a fim de alcançar os mesmos objetivos.

Os testadores que realizam os testes de integração precisam levar em consideração que o elemento testado é tão somente a integração. Ou seja, o teste de integração não se preocupa com a maneira como as funcionalidades foram desenvolvidas, nem verifica se cada funcionalidade ou parte do software funciona da maneira como havia sido projetada. Esse nível de teste procura identificar problemas na integração tanto das partes do próprio software quanto do software com outros sistemas com os quais tenha interdependência.

É preciso tomar algumas medidas para que os testes de integração sejam realizados com sucesso (DELAMARO; MALDONADO; JINO, 2016). Veja a seguir:

- preparar previamente um ambiente para que o teste aconteça;
- preparar previamente um banco de dados para servir de teste;
- registrar formalmente todos os resultados obtidos com os testes;
- avaliar todos os resultados obtidos por meio dos testes;
- relatar ao gerente de projeto toda ocorrência de problema mais grave que possa impactar negativamente o andamento do restante do projeto de desenvolvimento do software.

Os níveis de teste de aceitação e de regressão

Normalmente, o nível de teste de aceitação diz respeito à aceitação, por parte dos usuários finais, do software entregue. Já o teste de regressão se relaciona com uma nova validação e uma nova repetição de testes já feitos. A ideia é averiguar se modificações não acarretaram algum problema de execução que anteriormente não existia ou já havia sido corrigido.

Teste de aceitação

Nesse nível de teste de software, normalmente os testes são realizados por um grupo de amostra, com representantes dos usuários finais do sistema. Por esse motivo, esse nível de teste também pode ser chamado de teste de aceitação do usuário (MARTINS, 2007).

Durante a realização dos testes de aceitação, os usuários selecionados recebem a tarefa de simular todas as operações que o sistema deve executar de maneira rotineira. A ideia é que os usuários identifiquem se o comportamento apresentado pelo sistema está de acordo com o que foi solicitado e com o que foi levantado como requisito e necessidade inicial do sistema.

Fique atento

O teste de aceitação é responsável pela validação do software como um todo pelo usuário final, pelo cliente ou pelas pessoas que vão utilizá-lo de alguma maneira. Ele ocorre por meio da utilização de dados e cenários semelhantes aos reais.

Em outras palavras, o teste de aceitação representa um teste mais formal. Os usuários se envolvem nele de tal forma que possam determinar se o sistema funciona corretamente, da maneira esperada, e se ele satisfaz ou não os critérios de aceitação estabelecidos pelo cliente. É o teste de aceitação que determina se o cliente aceita ou não o sistema da forma como está sendo entregue.

O teste de aceitação do usuário é sempre realizado antes que o software seja implementado de fato, pois o real motivo de se realizar esse tipo de teste é exatamente entender se o software está pronto ou não, na visão dos usuários que vão utilizá-lo em sua rotina. Isso acontece porque, antes que o software seja instalado, implementado e dado como entregue, é preciso saber se os

usuários vão conseguir realizar todas as atividades, atingir todos os objetivos e satisfazer todas as necessidades que existiam no início do projeto de software.

Apesar de ser um tipo de teste, o teste de aceitação do usuário não é propriamente um teste realizado em busca de problemas e defeitos. O que se quer a partir de um teste de aceitação é estabelecer confiança e aumentar a segurança no que foi desenvolvido, seja em uma parte ou um módulo, ou no sistema como um todo. É comum encontrar na literatura outra noção a respeito do teste de aceitação, que é o teste de aceitação de entrega. Nesse outro entendimento, o teste de aceitação é caracterizado por uma entrega, e pela consequente aceitação, de uma equipe para outra durante a fabricação do software.

Exemplo

Considere o caso de uma equipe que desenvolveu determinado módulo de um software, que contém várias funcionalidades, e precisa entregar o que foi construído até o momento para outra equipe, que vai fazer a construção de um módulo diferente. Nesse caso, a equipe que está recebendo deve fazer um teste de aceitação para verificar se o que a equipe anterior realizou está a contento.

Ainda existe outro tipo de teste aceitação, que é aquele que acontece logo no início do projeto, quando o contrato é assinado. Ele se chama teste de aceitação de contrato e regulamento. No teste de aceitação de contrato, são verificadas as cláusulas contratuais de fabricação do software. Quando o teste é de aceitação de regulamento, é averiguada a necessidade de cumprimento de alguma legislação, acordo ou norma.

Teste de regressão

Normalmente, no nível do teste de regressão, ainda estão sendo desenvolvidas versões do software, ele ainda está sendo codificado. Esse é um tipo de teste aplicado sempre que uma nova versão da aplicação fica pronta. Ele também é aplicado quando é necessário realizar um novo ciclo de testes em toda a aplicação para verificar se elementos novos que foram desenvolvidos não afetaram de maneira negativa o restante das partes que já haviam sido codificadas. É por esse motivo que o teste de regressão pode ser chamado também de teste decorrente de mudança (MARTINS, 2007).

O teste de regressão leva em consideração que os testes feitos anteriormente precisam ser realizados novamente. O objetivo é verificar se partes novas de codificação não afetaram ou alteraram o comportamento do sistema dependendo das entradas que ele recebe. O teste de regressão é, via de regra, a realização de todos os testes novamente, como se fossem a sua repetição. A cada vez que o software sofre uma modificação, todas as suas partes são testadas novamente, a fim de identificar problemas.

Fique atento

O teste de regressão é realizado em um software que já foi testado depois que ele sofre qualquer tipo de modificação estrutural ou de interface. A ideia é verificar a existência de algum problema novo com relação à versão anterior do software, ou ainda o reaparecimento de um problema que já havia sido solucionado em outro ciclo de testes.

Uma grande desvantagem do teste de regressão é que ele é bastante demorado. Como você deve imaginar, isso pode impactar os custos e também causar o não cumprimento do prazo global do projeto. Para que esse teste seja vantajoso, deve ser planejado de maneira detalhada, com base em estratégias. Além disso, ele deve testar todas as funcionalidades críticas do software, de maneira repetida, a cada mudança de estrutura.

A mudança de estrutura é o principal elemento gerador do teste de regressão. Por isso, é importante conhecer quais são as principais causas de mudanças em estruturas de softwares. Veja a seguir.

- **Correção de erros:** normalmente exige a modificação do código-fonte dos programas e é possível que essa modificação acarrete outros erros.
- **Alteração de funcionalidade:** é o caso de o usuário solicitar modificações em funcionalidades já implementadas, ou a inclusão de novos aspectos em partes já desenvolvidas.
- **Desenvolvimento de nova funcionalidade:** esse é o motivo mais comum para os testes de regressão existirem. Sempre que uma nova funcionalidade é codificada, é importante que os desenvolvedores tenham em mente que ela deve ser totalmente compatível e integrável com a estrutura antiga.

Como resultado de um teste de regressão, pode-se chegar à conclusão de que surgiram problemas que ainda não existiam. Nesse caso, o teste de regressão considera que o sistema teve uma regressão, ou regrediu. Afinal, em vez de apresentar um funcionamento perfeito, ele sofreu um retrocesso. Esse é o caso também de os testes terem sido realizados e ressurgirem problemas que já haviam sido solucionados em outro ciclo de testes.

Dizer que um software regrediu não é nada animador para a equipe envolvida no projeto de desenvolvimento. Então, é preciso garantir que os testes de regressão encontrem código e interface intactos após as modificações sofridas no software.

Referências

DELAMARO, M. E.; MALDONADO, J. C.; JINO, M. *Introdução ao teste de software.* 2. ed. São Paulo: Elsevier, 2016.

MARTINS, J. C. C. *Técnicas para gerenciamento de projetos de software.* Rio de Janeiro: Brasport, 2007.

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