PLANEJAMENTO DA EXECUÇÃO DE TESTES DE SOFTWARES MANUAIS

Nailton Soares de Almeida Junior[1]
Edson Cruz de Lima Júnior[2]

RESUMO

O presente artigo trata-se de um estudo sobre as fases do planejamento do teste de software. A pesquisa foi aplicada através da observação direta de uma empresa de tecnologia que desenvolve programas para computador, onde foi analisado os processos executados pela equipe de teste existente na companhia. Esse estudo científico tem como objetivo geral expor as melhores práticas no planejamento de testes de softwares manuais, que venham impactar de forma positiva na execução dos mesmos. Para o embasamento do presente trabalho foi utilizado de leituras sobre metodologias e técnicas para a implementação de um processo formal de testes. Por via de um estudo de caso feito, foi identificada falhas vindas de uma má preparação dos testes, que acabavam gerando desempenho insatisfatório nas validações no software. Falta de recursos ou alocação de profissionais sem experiências técnicas em atividades complexas foram algumas das situações negativas constatadas. A partir das investigações diárias realizadas sobre o projeto de software da empresa, foi reconhecida uma falta de qualidade no planejamento do teste, e a necessidade de mudança para uma visão mais crítica desde o levantamento de escopo até propriamente a fase de execução. Para tanto, foi elencada possíveis soluções para serem implementadas afim de evitar e/ou mitigar riscos de projeto futuros.

Palavras-chave: Planejamento de Teste; Qualidade; Teste de Softtware.

[1] Graduado em Análise e Desenvolvimento de Sistemas – UTP.
[2] Professor Orientador, Mestre profissional em Engenharia de Software – CESAR.EDU.

1 INTRODUÇÃO

O teste de software é uma área da tecnologia que atua no processo de vida do desenvolvimento de programas, onde possui profissionais na maior parte das vezes alocados na iniciativa privada, e também em órgão públicos, mas em menor escala. Tem como objetivo encontrar erros no produto e assegurar a qualidade de sistemas computacionais, validando se os requisitos para uma determinada aplicação estão sendo entregues como esperado. As validações são feitas desde em documentos produzidos no processo de desenvolvimento do programa, como requisitos funcionais e fluxogramas, até no software propriamente criado pelos desenvolvedores.

Os profissionais desta área comumente chamados de testadores, tinham a função de forma exploratória identificar falhas no software que eram geradas a partir de defeitos no código fonte do programa, mas com o amadurecimento das equipes de tecnologia, foi percebida a necessidade da criação e uso de metodologias, para que o processo de teste seja mais confiável a partir de um planejamento coerente das atividades do teste e a execução de provas no software, e não somente uma pessoa utilizar um programa de forma aleatória, sem contexto e objetivo do teste.

Mesmo que diversos profissionais tenham percebido a necessidade da qualidade em seus projetos de software, e novas técnicas para a realização de teste tenham sido criadas para suportar um processo mais conciso de verificações, infelizmente muitos aplicativos são entregues com atraso, pois houveram falhas na fase de execução do teste provenientes de um planejamento mal feito. Se não forem levados em consideração os custos, prazos, recursos e riscos para a realização do teste de forma correta, poderão ocorrer perdas financeiras, atritos na confiança entre o fornecedor e o cliente e ações legais por não cumprimento de obrigações contratuais.

Ao pensar nos danos mencionados acima, torna-se relevante analisar as melhores de formas de mitigar inconsistências que venham impactar o desempenho do teste a partir do planejamento do mesmo.

O presente artigo permitirá uma investigação científica, calcada em pesquisas bibliográficas e estudos de caso a partir de observação *in lócus*, onde foi obtido dados de uma equipe de testes de software alocada em um projeto de uma empresa

implementadora de programas de computador. O nome da empresa como de seus respectivos funcionários e clientes não serão expostos neste trabalho por questões de regras internas de privacidade da organização. A análise foi realizada para contemplar o objetivo geral de expor melhores práticas no planejamento de testes de softwares manuais, que venham impactar de forma positiva na execução dos mesmos, e possui os seguintes objetivos específicos: 1 - identificar as deficiências de uma má preparação dos testes, que geram um baixo desempenho na execução diária de cenários de teste; 2 - recomendar práticas para serem aplicadas no planejamento dos testes que poderão otimizar o tempo de execução dos testes.

2 A QUALIDADE NO PLANEJAMENTO DO TESTE

Segundo Satpathy (2017, p. 86), “qualidade é a capacidade dos produtos ou entregas concluídas em atender os critérios de aceitação e em alcançar o valor de negócio esperado pelo cliente”. Mesmo que este conceito esteja associado aos entregáveis finais de um determinado serviço para com o seu contratante, é necessário salientar que a qualidade deverá ser priorizada desde o início. É possível associar que a qualidade está atrelada a ações preventivas contra riscos e falhas, já que o custo para a correção de inconsistências manifestadas em um determinado produto ou serviço é muito mais alto nos momentos finais de um projeto.

Glenford J. Myers, um dos maiores “gurus” da área de qualidade e teste de software, em seu livro The Art of Software Testing, publicado em 1979, já dizia que os custos para a correção de um erro tendem aumentar em dez vez mais a cada fase do projeto de software.

Figura 1: Gráfico exemplificando regra 10 de Myers

Fonte: site DevMedia (2019)

Ainda observando a regra “10 de Myers”, pode-se dizer que o custo para a correção de um defeito ainda pode ser afetado de acordo a complexidade e criticidade da mesma, já que possivelmente será necessário que sejam adicionadas mais horas na jornada de trabalho da equipe e até alocações extras de recursos(profissionais) ocorram para o desenvolvimento de uma solução.

De acordo o que foi dito anteriormente é possível alegar que a qualidade é sinônimo de economia orçamentária, com a probabilidade da diminuição de retrabalhos proveniente de erros e seus respectivos gastos adicionais atrelados. Mas como podemos fazer com que as equipes associadas em projetos, empreguem qualidade em suas atividades diárias? Para Crosby (1990) é necessário que cada profissional tenha um senso de qualidade próprio como propósito para a produção de um trabalho sem falências, e que todo esforço tenha um enfoque mais pessoal, como um objetivo particular a ser atingido. Com essa ideia, fica interpretável que a adição ou melhoria da qualidade em um processo, precisa ser tratada de forma cultural e não só uma imposição da burocracia da empresa, e indo pelo caminho do viés “A qualidade começa em mim”, que deverá estar internalizado em cada pessoa da organização.

Para Bond, Busse e Pustilnick (2012), de forma crescente, a qualidade vem sendo o interesse de diversos setores da economia, e a mesma está cada vez mais mutável e adaptativa de acordo as necessidades encontradas. Focando a afirmação acima na área da tecnologia, é possível identificar à vontade em utilizar a qualidade para se atingir a eficiência e eficácia em produtos que serão entregues por parte de seus profissionais.

Analisando a empresa de produção de software que foi utilizada como estudo de caso deste artigo, percebeu-se a real busca pela qualidade em aplicativos, mas infelizmente foi observado que a responsabilidade para evitar a ocorrência de falhas bem como manter o nível de qualidade dentro dos limites definidos, recai somente sobre a equipe de testes de software. Como a qualidade de fato não foi difundida por toda organização de forma cultural, mas sim imposta como uma exigência pelo alto escalão (diretores) da empresa, a equipe, precisa garantir a ausência de erros no produto provenientes de outras fases do projeto como a análise de requisitos, criação das especificações e a própria produção do software em si. Neste contexto, inicialmente não foi identificada falhas que impactam diretamente na qualidade do programa no fim do projeto, mas com o passar das semanas de levantamentos de

dados obtidos de forma exploratória direta, foi constatado que a centralização da qualidade na equipe de teste estava criando somente a precaução de entregar softwares sem defeitos, deixando de lado o conceito de melhoria contínua na totalidade desta fase do projeto, ou seja, as atividades exercidas pelos testadores estavam sendo feitas de forma mecanizada sem haver qualquer indagação se a maneira que o teste está sendo conduzido possui um bom desempenho ou ocorre a necessidade de melhorar pontos da execução.

Com o aspecto mencionado acima, foi questionado aos gerentes envolvidos, se as atividades conduzidas pelos testadores atingiam as expectativas do projeto, as respostas mais ditas eram que os testes possuíam uma boa cobertura das funcionalidades do software e poucas falhas eram manifestadas no ambiente de produção do cliente, já que os defeitos eram identificados pelos respectivos profissionais, mas ocorriam diversos atrasos para a finalização das provas, já que a execução dos testes não seguiam o planejamento feito anteriormente, levando a um atraso na entrega do software para o contratante.

3 FASES DO PLANEJAMENTO DE TESTES, SUAS POTENCIALIDADES E LIMITAÇÕES

De acordo com a instituição Internacional Software Testing Qualifications Board (ISTQB) (2018, p. 75):

> O planejamento do teste é influenciado pela política de teste e a estratégia de teste da organização, pelos ciclos de vida e métodos de desenvolvimento pelo escopo dos testes, objetivos, riscos, restrições, criticidade, testabilidade e disponibilidade dos recursos.

Tendo em consideração a menção acima, é capaz de se fazer um paralelo entre a qualidade e o planejamento dos testes, onde que ambos são altamente influenciados por diversos fatores que necessitam serem levados em consideração na fase inicial e no decorrer de um projeto de software.

Se o planejamento não foi realizado de forma correta, sem mensurar os diversos cenários de riscos e as fraquezas do projeto, infelizmente é provável que durante a execução dos testes ocorram equívocos provenientes de um escopo mal feito.

3.1 O escopo do teste

Para ser feito o planejamento do teste, a primeira atividade a ser realizada é o levantamento de escopo, onde que a partir desta análise, será obtida uma lista que irá documentar todos os objetivos e necessidades para a posterior execução dos testes. É crucial que neste momento seja identificado o que e como será testado o software. Como dito por Olsen (2018), é impossível realizar testes para todas as opções de um sistema, exceto em casos triviais. Desta forma a priorização dos testes sobre módulos mais utilizados pelos usuários e/ou mais críticos se torna extremamente necessária. Podemos elencar os seguintes marcos que serão identificados nesta fase: o prazo (tempo) para estes testes serem feitos, quantidade de testadores que serão alocados para fazer as devidas provas e as especificações para preparação do ambiente de teste.

Durante o estudo de caso, foi observado que muitos dos problemas que foram apresentados durante a execução do teste foram provenientes desta fase, pois ocorreram negligências nas métricas e resultados esperados sobre os testes que seriam feitos pela equipe.

A primeira inconsistência identificada foi a criação de cronograma absurdamente curto, onde a equipe composta por quatro pessoas deveria executar em apenas três semanas mais de 1000 casos de testes, sendo que possuíam uma complexidade alta e diversas interdependências. O resultado mais impactante deste cenário foi o atraso da entrega do projeto, pois os testadores em menor número não conseguiram executar todas as suas pendências no tempo limite. Em uma situação deste tipo, se quantidade total de testes definidos não pode ser diminuída, há duas soluções possíveis a serem sugeridas, é negociado com o cliente um período maior para execução de todos os testes por uma equipe reduzida ou é aumentado o número de recursos para que toda a cobertura de verificações almejada seja atingida.

A segunda imprecisão foi resultante da não aplicação das soluções mencionadas acima, onde que para compensar o curto prazo e um baixo número de funcionários empregados no projeto, foi aumentada a quantia de testes a serem executados por dia para cada pessoa. Inicialmente o volume de provas feitas estavam atingindo as expectativas e algumas vezes até superando-as, mas depois de alguns dias o excesso de trabalho gerou um desgaste da equipe e isso comprometeu a qualidade dos testes. A falta de atenção proveniente do cansaço físico e mental sobre os testadores fez com que muitas das vezes os testes fossem executados de forma incorreta, não condizente com o que era aguardado nos resultados de saída definidos, e por sua vez as evidências (capturas de tela) criadas para documentar as execuções refletiam os mesmos erros. Além desta falha técnica, as métricas do projeto também eram afetadas, pois quando a meta dos testes não era alcançada, uma nova estimativa era feita somando os testes do dia atual com os que não foram executados da data anterior, aumentando em muito o porcentual de atrasos no monitoramento do projeto de teste e fazendo com que o cliente duvidasse da capacidade da empresa contratada de entregar o produto com a qualidade almejada no prazo estipulado.

Também na empresa observada em questão, foi visto que os *testers* (testadores) ministravam treinamentos de como usar os sistemas para os usuários internos do cliente. Outra inadequação foi identificada nesta fase, no qual o escopo continha um paralelismo de atividades, onde que durante o momento da execução dos testes era previsto que também ocorressem os treinamentos mencionados. Em consideração a conjuntura de falta de prazo, mínima alocação de recursos e quantidade alta de casos de testes para verificação, pode-se notar que foi impraticável a execução de duas tarefas distintas ao mesmo tempo, gerando ainda mais atrasos. Ao fim dessa análise sobre o escopo do teste, foi repassado para os gerentes da organização que a melhor solução a ser aplicada para os erros mencionados acima é a mudança de uma visão super otimista para uma mais realista, já que na criação do escopo é necessário identificar os pontos fracos existentes e não ignorá-los, achando que a equipe de testes independentemente da situação conseguirá entregar os resultados calculados e aguardados.

Uma forma de adequar o escopo do teste a realidade da empresa é a utilização da relação tripla de qualidade exposta por MONTES (2017).

Tabela 1 – Adequação do projeto e o custo da mudança

Mudança desejada	**Ação a ser tomada**
Aumento de escopo	Aumenta o custo e/ou prazo
Diminuir o prazo	Aumenta o custo e/ou reduz o escopo
Diminuir o custo	Reduz o escopo

Fonte: Elaborado pelo autor (2019)

Figura 2: Relação tripla de qualidade apresentando os marcos que são considerados

Fonte: site Arco Pleno (2019)

3.2 Alocação de recursos

Após a definição de escopo e identificada a quantidade de recursos que deverão ser utilizados, é necessário que seja elegido quais profissionais farão a escrita e execução dos casos de testes. Para ISTQB (2014, p. 68) caso de teste é:

> Conjunto de valores de entrada, precondições de execução, resultados esperados e pós-condições de execução desenvolvidas para um determinado objetivo ou condição de teste, tais como para exercitar o caminho de um determinado programa ou verificar o atendimento a um requisito especifico. (Tradução Brazilian Software Testing Qualification Board).

Com o significado acima e sabendo que os sistemas estão se tornando cada vez mais complexos e possuem diversos fluxos que refletem as regras de negócios das organizações, podemos captar que para cada tipo de caso de teste, é necessária senioridades e experiências distintas dos testadores alocados.

Na observação direta feita, foi validado que os profissionais de teste da empresa analisada, não conheciam todos os módulos da cadeia de software ofertada como produto para clientes externos, mas sim, havia especialistas em certos programas, como o especialista em aplicativos para controle de estoque, gerenciamento de preços de mercadorias e etc. Nesta circunstância, quando um *tester* era designado para escrever um caso de teste de uma função que o mesmo não tinha conhecimento, muita das vezes era deixado de ser considerado variáveis que eram relevantes para o mantimento da qualidade. A mesma situação acontecia no momento da execução dos casos, onde que a pessoa que iria executar o teste acabava por falta de referência técnica negligenciando particulares que deveriam ser respeitadas em determinados software, fazendo só uma validação superficial e gerando evidências pobres do teste realizado.

Para evitar ocorrências deste tipo, primeiramente é básico avaliar a experiência de cada profissional para que seja designado atividades que venham corresponder com o nível de competência de cada um. Em seguida é possível que dentro da equipe, fisicamente sejam posicionados testadores menos experientes com os que possuem mais expertise metodológica de testes e do software, para que seja incentivado a troca de informações e o abismo de conhecimento seja diminuído entre as partes.

3.3 Escrita dos casos de teste

Posteriormente a distribuição das atividades de acordo com a capacidade técnica de cada testador, é iniciada a fase de criação dos casos de testes. Com a ideia que um teste mal escrito será um teste mal executado, podemos começar a analisar a necessidade da escrita de bons casos de testes para que os mesmos verifiquem de forma correta as funções do software e também contemplem a qualidade esperada.

Crosby (1990) disse que requisitos devem ser feitos para que gerem fácil compreensão e implementação por qualquer pessoa que os tenha em um projeto. Expandindo este conceito para os casos de testes, e levando em consideração que dentro de uma equipe há profissionais com diferentes níveis de experiência em tecnologia, como vimos no subcapítulo anterior, é necessário que seja assegurado que independentemente da competência de cada testador, o teste poderá ser feito.

Se na fase de execução, a equipe de teste faz a leitura dos casos e coloca em prática o que foi definido no planejamento, é no momento da criação dos testes que deverá ser identificado como será testado o programa. Ao realizar a escrita das provas, é importante que seja levado em consideração o nível de detalhamento das validações. Se os *testers* não conhecem o sistema e ocorre uma grande rotatividade de profissionais na empresa, como foi averiguado na análise feita, é recomendado que haja um certo detalhamento das ações a serem executadas futuramente no software, para que não ocorra atrasos na fase de execução devido à falta de informações.

Figura 3: Caso de teste sem detalhamento, criado no programa Microsoft Excel

Caso de teste	Sistema	Descrição	Resultado Esperado
Cadastro de fornecedor	Fornecedor Link	Cadastre um fornecedor no sistema	Fornecedor cadastrado

Fonte: Elaborado pelo autor (2019)

Figura 4: Caso de teste detalhado, criado no programa Microsoft Excel

Caso de teste	Sistema	Criticidade	Descrição	Resultado Esperado
Cadastro de fornecedor	Fornecedor Link	Baixa	Acesse o sistema Fornecedor Link pela url https://fornecedorlink.com.br	Página do sistema carregada
			Realize login com credências de segurança	Login realizado
			No menu esquerdo de ações, clique em "Fornecedor > Cadastro de Fornecedor"	Janela para cadastro de fornecedor é exibida
			Nos campos obrigatórios marcados com asterisco(*) vermelho, informe os dados do novo fornecedor	Informações do fornecedor são exibidas nos campos do formulário
			Clique no botão "Salvar"	Janela de cadastro será fechada
			Valide se o novo fornecedor foi salvo executando o comando no banco de dados : *select * from supplier where id = id_do_novo_fornecedor*	Consulta no banco de dados retorna o novo fornecedor

Fonte: Elaborado pelo autor (2019)

Comparando as imagens acima, fica claro que casos mais detalhados auxiliam testadores que não sabem como utilizar um determinado aplicativo, exatamente por orientar o profissional a executar os passos necessários para que o teste seja finalizado.

Um outro ponto a ser considerado é que as instruções do caso de teste devem condizer com o resultado de saída almejado. Se o testador executa ações errôneas vindas de um caso de teste mal projetado, consequentemente haverá desvios no fim das validações. À vista disso, é essencial que sejam estabelecidos passos e indicações que retornem evidências relevantes para o teste.

3.4 Priorizando a execução dos casos de teste

Com a devida criação dos casos, é crucial que saibamos a ordem que os testes serão executados. Se levarmos em consideração que um sistema possui tanto funções básicas como o cadastro de um usuário, quanto mais críticas que podem causar danos à vida ou perdas financeiras, devemos criar uma priorização dos testes mais importantes. Uma abordagem a ser utilizada é a adição de severidades para cada teste, como “Alta”, “Média” ou “Baixa”, onde planejamos a execução dos testes com prioridades mais altas e em seguida as mais baixas, respeitando assim os módulos que mais impactam o negócio do cliente.

Para auxiliar na identificação da correta severidade de cada caso de teste, podemos fazer uma análise de risco que consiste em avaliar a probabilidade e o impacto de um evento negativo ocorrer, que posteriormente poderá gerar uma falha no sistema e irá afetar a qualidade do mesmo. Também é imprescindível manter a comunicação aberta com os usuários que irão utilizar o sistema e partes interessadas nesta fase, para que todos possam contribuir com suas opiniões a respeito do que é crítico a ser verificado, evitando que somente a equipe de testadores façam a priorização dos testes e as necessidades do cliente não venham ser atendidas sobre o ponto de visto de negócio.

4 CONSIDERAÇÕES FINAIS

Esse estudo viabilizou a concepção que a qualidade não deverá está empregada somente no produto final, mas sim em todas as fases de um projeto, e neste caso, principalmente no próprio planejamento dos testes, para que ocorra uma execução de provas imprecisa.

A fase de testes de um determinado software não deve ser uma mera formalidade de um contrato previamente assinado ou vista como uma bondade da empresa contratada em fazer uma avaliação do seu produto para o seu cliente, mas sim, ser a última garantidora de qualidade do projeto. Levando em consideração que na produção de um programa de computador ou aplicativo de celular poderão ocorrer falhas sistêmicas, já que programadores são pessoas passíveis de cometer erros, é muito importante que seja aprendido como conter essas inconsistências na fase de testes e que os requisitos esperados estejam em conformidade.

É razoável o entendimento que, o maior problema para as condições mencionadas neste artigo, como atrasos em entregáveis e execuções mal feitas de testes, é a criação de um escopo com expectativas fantásticas que são inviáveis de serem conquistadas. Desta forma podemos afirmar que desde o momento da venda do produto para um cliente, é necessário que haja em mente quais pontos fracos e riscos que aquele projeto possui, junto com suas respectivas soluções, para que acordos negociados anteriormente não venham ser quebrados por falta de estimativas “edificadas” no bom senso profissional que toda empresa de tecnologia deveria possuir.

5 REFERÊNCIAS

BOND, Maria Thereza; BUSSE, Angela; PUSTILNICK, Renato. **Qualidade total: o que é e como alcançar**. Curitiba: Editora Intersaberes, 2012.

CROSBY, Philip. **Qualidade Falando Sério**. São Paulo: Editora McGraw-Hill Ltda, 1990.

INTERNACIONAL SOFTWARE TESTING QUALIFICATIONS BOARD. **Certified Tester Foundation Level Syllabus**. [S. l.: s. n.], Edição BR, 2018, p.75. Disponível em: < https://www.bstqb.org.br/uploads/syllabus/syllabus_ctfl_2018br.pdf>. Acessado em 15/01/2019.

INTERNACIONAL SOFTWARE TESTING QUALIFICATIONS BOARD. **Standard Glossary of Terms used in Software Testing**. [S. l.: s. n.], version 3.1, 2014, p. 68. Disponível em < https://astqb.org/resources/glossary-of-software-testing-terms/>. Acessado em 15/01/2019.

MONTES, Eduardo. **Introdução ao Gerenciamento de Projetos**. São Paulo: Create Space Independent Publishing, 2017, p.51.

MYERS, Glenford.**The Art of Software Testing, Second Edition**. New York: Word Association, 1979.

SATPATHY, Tridibesh**. Guide to the scrum body of knowledge (SBOK™GUIDE) 3rd Edition**. Arizona: VMEdu, 2017, p. 86.