MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO TRIÂNGULO MINEIRO

Métricas de Software

Barbara Ramos Alves
Diego Henrique Martins Diniz
Lais Danielle Santos Vieira
Milena Arantes Bertolini

Uberlândia

2024

1. INTRODUÇÃO

Métricas de software são medidas quantitativas que ajudam a quantificar diferentes aspectos do desenvolvimento de software, do processo de desenvolvimento em si ao produto resultante. Elas são usadas para avaliar a qualidade, o custo, o prazo, a eficiência e o desempenho do software, bem como para ajudar a identificar áreas para melhoria. Utilizar métricas é uma prática fundamental para uma boa gestão.

Uma boa métrica é aquela que permite a construção de indicadores que facilitam a tomada de decisão sem que sua confiabilidade seja questionada. Essas por si só, podem não prover respostas imediatas, mas os indicadores oriundos de métricas sólidas podem dar um indício do que está ocorrendo. Comparando os indicadores com informações e experiências anteriores, se têm o potencial de dar uma visão do que pode vir a ocorrer no futuro. Mesmo que uma previsão tenha uma certa margem de erro, ela limita um conjunto de possibilidades e isso é o que se deseja ter para que se possa atuar dentro de um projeto.

2. MÉTRICAS DE TESTE DE SOFTWARE

As métricas de testes de software são medidas quantitativas que ajudam a avaliar a eficácia, a eficiência e a qualidade dos testes realizados durante o processo de desenvolvimento de software. Essas métricas são usadas para quantificar diferentes aspectos dos testes, fornecendo percepções sobre a cobertura de testes, a qualidade do código testado e a eficiência dos processos de teste.

A criação de testes de forma manual ou automatizada, é uma das principais atividades empregadas para melhorar a qualidade de produtos de software. Uma das maneiras de coletar dados para a geração de métricas é a análise de testes que foram aprovados, testes que falharam, testes executados, testes bloqueados, entre outras. Essas informações possibilitam analisar o nível de confiabilidade do sistema e apontam ações a serem realizadas para a melhoria do processo de teste, além de contribuir com a tomada de decisão em relação ao andamento dos projetos de controle da qualidade de software.

As métricas de teste se dividem em duas outras métricas denominadas básicas e derivadas:

- Métricas básicas: são informações adquiridas de forma direta a partir da aplicação dos testes, sem intermediários ou processos adicionais. São usados para acompanhamento da situação e da evolução do projeto. Como exemplos têm-se a quantidade de casos de testes criados, executados, que passaram, que falharam, entre outros.
- Métricas derivadas: são obtidas através de processos adicionais mediante a conversão das métricas básicas em dados mais úteis que, combinados, podem ser utilizados para avaliar mudanças no processo. Como exemplos têm-se o percentual dos testes concluídos, da cobertura dos testes, dos casos de teste que passaram ou que não passaram, dos defeitos, dos defeitos corrigidos, da efetividade e da eficiência dos testes, a taxa de defeitos descobertos e o custo de remoção dos defeitos.

Para utilizar as métricas de teste de forma eficaz, é fundamental considerar a complexidade do sistema que será testado.

3. TIPOS DE MÉTRICAS DE TESTE

Existem diversos tipos de métricas de testes de software, que podem ser classificadas de várias formas. Algumas dessas métricas podem ser coletadas e analisadas utilizando ferramentas específicas, como JUnit, TestNG, Selenium, Jenkins, entre outras. Essas ferramentas permitem a automação da execução de testes e a geração de relatórios detalhados, facilitando a coleta e análise de métricas de teste de software em projetos de desenvolvimento. Por exemplo, o JUnit e o TestNG são amplamente utilizados para testes unitários, enquanto o Selenium é comumente empregado para testes de interface do usuário. O Jenkins, por sua vez, pode ser utilizado para integração contínua e automação de testes, fornecendo informações valiosas sobre o desempenho dos testes ao longo do tempo.

- Total de Defeitos Detectados (TDD): avalia a qualidade do software com base na quantidade de defeitos encontrados durante os testes em relação a uma unidade de medida, geralmente o tamanho do código fonte.
- Total de Defeitos Encontrados pelo Cliente (TDC): medida que avalia a qualidade do software com base na quantidade de defeitos relatados pelo cliente após o software ter sido entregue ou colocado em produção.
- Total de Defeitos Removidos (TDR): medida que avalia a eficácia dos processos de teste e revisão de código em encontrar e corrigir defeitos, ou seja, avalia a quantidade de falhas que foram removidas, após a identificação dos erros.
- Eficácia na Detecção de Defeitos (EDD): medida que avalia a capacidade dos testes de encontrar defeitos no software. É calculada com base na quantidade de defeitos encontrados durante os testes em relação à quantidade total de defeitos presentes no software.
- Tempo Médio de Reparo (MTTR): medida que mostra quanto tempo, em média, se leva para que a equipe consiga identificar os erros no sistema e corrigi-los. Quanto menor for o tempo médio de reparo, mais eficiente é a equipe. Auxilia a garantir que os problemas sejam solucionados em um tempo razoável.
- Tempo Médio Entre Falhas (MTBF): medida que indica os intervalos de tempo entre falhas.
- Taxa de sucesso da resolução de defeitos: medida que quantifica e faz uma relação entre o total de falhas solucionadas e a reincidência.

4. EXEMPLO DE MÉTRICAS DE TESTE

1. Taxa de Cobertura de Código:

Esta métrica indica a porcentagem do código-fonte do software que foi exercida pelos testes. Uma alta taxa de cobertura de código geralmente sugere uma maior

confiabilidade do software. Por exemplo, em um projeto com 1000 linhas de código, se 800 linhas forem testadas, a taxa de cobertura de código seria de 80%.

2. **Taxa de Cobertura de Testes Unitários:**

Esta métrica mede a porcentagem de testes unitários executados com sucesso em relação ao total planejado. Testes unitários bem-sucedidos são fundamentais para garantir a integridade das unidades individuais de código. Por exemplo, se 45 dos 50 testes unitários planejados passarem com sucesso, a taxa de cobertura de testes unitários seria de 90%.

3. **Tempo Médio de Execução de Testes:**

Esta métrica fornece uma visão do tempo médio necessário para executar um caso de teste. O tempo de execução é crucial para estimar os recursos necessários e otimizar o processo de teste. Por exemplo, se 100 casos de teste foram executados em um total de 5000 segundos, o tempo médio de execução por caso de teste seria de 50 segundos.

4. **Taxa de Defeitos por Caso de Teste:**

Esta métrica indica a média de defeitos encontrados por caso de teste. É uma medida importante da qualidade do software e da eficácia dos testes. Por exemplo, se 20 defeitos foram encontrados em 100 casos de teste, a taxa de defeitos por caso de teste seria de 0.2.

5. **Taxa de Reteste:**

Esta métrica avalia a eficácia da correção de defeitos, mostrando quantos defeitos foram corrigidos e retestados com sucesso em relação ao total de defeitos encontrados. Por exemplo, se 15 dos 20 defeitos encontrados foram corrigidos e passaram no reteste, a taxa de reteste seria de 75%.

6. **Taxa de Aceitação de Defeitos:**

Esta métrica indica a proporção de defeitos encontrados que foram aceitos pelo cliente após correção. É fundamental para medir a satisfação do cliente e a qualidade percebida do produto. Por exemplo, se apenas 10 dos 20 defeitos

encontrados foram aceitos pelo cliente após a correção, a taxa de aceitação de defeitos seria de 50%.

Além dos exemplos previamente apresentados, é importante considerar cenários adicionais para ilustrar como as métricas de teste de software podem ser aplicadas em diferentes contextos. Por exemplo, imagine um projeto de desenvolvimento de um aplicativo móvel, onde a taxa de reteste é uma métrica crucial para garantir a estabilidade do aplicativo em diferentes dispositivos e sistemas operacionais. Nesse caso, a taxa de reteste pode ser calculada em termos de porcentagem de defeitos corrigidos e retestados com sucesso em relação ao total de defeitos encontrados em diferentes versões do aplicativo.

5. DESAFIOS NA UTILIZAÇÃO DE MÉTRICAS DE TESTE DE SOFTWARE

Apesar dos benefícios das métricas de teste de software, alguns desafios comuns surgem ao utilizá-las. Um desses desafios está na seleção adequada de métricas, onde é crucial identificar aquelas mais relevantes para um projeto específico. Outro desafio está na interpretação dos resultados, pois as métricas por si só não fornecem uma imagem completa e é necessário entender o contexto em que foram obtidas. Além disso, garantir a precisão dos dados é fundamental, o que pode ser alcançado através da automação da coleta e análise de métricas utilizando ferramentas especializadas. Esses desafios exigem colaboração em grande escala entre as equipes de desenvolvimento e seus interessados, a fim de garantir que as métricas de teste de software sejam utilizadas de forma eficaz para melhorar a qualidade do produto final.

6. CONCLUSÃO

Ao longo deste artigo, exploramos a importância das métricas de teste de software e discutimos algumas das principais ferramentas disponíveis para coletar e analisar essas métricas. Observamos como o uso adequado dessas ferramentas pode fornecer insights valiosos para melhorar a qualidade do produto final.

É evidente que as métricas de teste desempenham um papel crucial na garantia da qualidade do software. Elas permitem que as equipes de desenvolvimento identifiquem áreas de melhoria, avaliem o progresso ao longo do ciclo de desenvolvimento e tomem decisões informadas para otimizar os processos de teste.

Além disso, a utilização de métricas de teste pode levar a uma maior transparência e comunicação dentro da equipe, facilitando a colaboração e o compartilhamento de melhores práticas.

Para futuras pesquisas ou desenvolvimentos na área, sugerimos explorar ainda mais a automação e integração de ferramentas de coleta e análise de métricas. Além disso, é importante investigar abordagens inovadoras para interpretar e visualizar os resultados das métricas, tornando-os mais acessíveis e significativos para uma variedade de partes interessadas, desde desenvolvedores até gerentes de projeto.

Em resumo, as métricas de teste de software são uma ferramenta poderosa para impulsionar a qualidade e eficiência do desenvolvimento de software. Ao integrar práticas de medição e análise em nossos processos de teste, podemos garantir que entregamos produtos de alta qualidade que atendam às expectativas dos clientes e usuários finais.

7. REFERÊNCIA BIBLIOGRÁFICA

JUNIOR, José. Introdução a métricas de software. DevMedia, 2016. Disponível em: <https://www.devmedia.com.br/introducao-a-metricas-de-software/36856>. Acesso em: 28 de fevereiro de 2024.

PECA, Sara. Vamos falar sobre métricas de testes e desenvolvimento?. One day testing, 2022. Disponível em: <https://blog.onedaytesting.com.br/metricas-de-testes-e-desenvolvimento/>. Acesso em: 28 de fevereiro de 2024.

BAUMGARTNER, Cristiano. 4 métricas essenciais para serem analisadas em relatórios de teste. Testing Company, 2022. Disponível em: <https://testingcompany.com.br/blog/4-metricas-essenciais-para-serem-analisadas-em-relatorios-de-teste>. Acesso em: 28 de fevereiro de 2024.

JUnit. Disponível em: https://junit.org/. Acesso em: 28 de fevereiro de 2024

TestNG. Disponível em: https://testng.org/. Acesso em: 28 de fevereiro de 2024

Selenium. Disponível em: https://www.selenium.dev/. Acesso em: 28 de fevereiro de 2024

Jenkins. Disponível em: https://www.jenkins.io/. Acesso em: 28 de fevereiro de 2024
Bamboo. Disponível em: https://www.atlassian.com/software/bamboo. Acesso em: 28 de fevereiro de 2024

SonarQube. Disponível em: https://www.sonarqube.org/. Acesso em: 28 de fevereiro de 2024