GESTÃO E QUALIDADE DE SOFTWARE - ATIVIDADE 02

Integrantes do grupo:
Gabriel Viegas Capecci – 82213442
Gabriel Mariotti Higa – 822141216
Gabriela Dardis Rodrigues – 822141330
Igor Britto - 822141647
Maria Fernanda Mendes Tobias – 822137255
Renato Peduto Filho - 822126254

São Paulo

2025

1. Impacto de Defeitos de Software nos Custos

A revisão de Software é algo de extrema importância para que possamos atingir um nível de qualidade de Software elevado aos “StakeHolders” envolvidos em toda a trajetória de criação do processo. Assim, temos que analisar todo um cenário para identificar quais os conceitos e técnicas de revisão de um Software podem afetar tanto positivamente quanto de maneira negativa o produto final.

Assim, ao analisarmos com cuidado quais os impactos que os defeitos de Software podem ter nos custos, é possível identificar:

- **Elevação dos custos operacionais:** Softwares de baixa qualidade podem gerar custos operacionais elevados, exigindo suporte contínuo, correções frequentes e atualizações constantes. Garantir qualidade desde o desenvolvimento inicial contribui para a redução de despesas a longo prazo e aprimora a eficiência dos processos;
- **Impacto nos resultados financeiros provenientes dos “StakeHolders”:** A insatisfação dos clientes pode levar ao abandono de um produto ou serviço, afetando diretamente a saúde financeira da empresa, por uma receita perdida contínua.

2. Amplificação e Eliminação de Defeitos

Para que possamos evitar os cenários descritos anteriormente é necessário que amplifiquemos a busca pela eliminação dos defeitos presentes em um Software. Este não é um processo simples, porém podemos seguir alguns processos para que possa ser feito com uma certa facilidade e assim padronizar essas ações para quaisquer softwares que forem desenvolvidos no futuro. Os pontos que podemos efetuar são:

- **Revisão de Código:** A verificação periódica do código é uma das estratégias mais eficazes para prevenir falhas. A revisão entre desenvolvedores permite a identificação precoce de erros, aumentando as chances de corrigir problemas antes que causem impactos significativos;

- **Testes Automatizados:** É fundamental para assegurar que as funcionalidades do software continuem operando corretamente, mesmo após mudanças no código;
- **Integração Contínua:** Isso permite a rápida identificação de conflitos e falhas, facilitando correções imediatas e evitando que erros se propaguem;
- **Design e Arquitetura Sólidos:** Um projeto bem estruturado e uma arquitetura robusta minimizam a ocorrência de falhas. O uso de princípios como a programação orientada a objetos e padrões de design contribui para um código mais organizado, modular e confiável.

3. Análise de Métricas de Revisão de Código

A análise de métricas de revisão de código avalia a qualidade e eficiência do processo de revisão, identificando gargalos e oportunidades de melhoria. Diferente da análise de defeitos, essas métricas ajudam a entender como as revisões impactam a colaboração, qualidade do código e velocidade de entrega.

3.1 Principais Métricas de Revisão

- **Tempo de Captação:** Mede o tempo desde a abertura de um pull request até o início da revisão. Um tempo elevado pode indicar atrasos e gargalos no processo;
- **Tempo de Revisão:**Avalia a duração da revisão de código. Revisões muito rápidas podem comprometer a qualidade, enquanto revisões longas podem retardar entregas;
- **Profundidade de Revisão:** Mede a média de comentários por revisão de pull request. Um maior número de comentários pode indicar revisões mais rigorosas, enquanto poucos comentários podem sugerir revisões superficiais;
- **PRs Mescladas sem Revisão:** Analisa a quantidade de pull requests aprovados sem revisão adequada. Uma alta taxa pode indicar falhas no controle de qualidade;
- **Taxa de Defeitos Pós-Revisão:** Mede a frequência de bugs identificados após a revisão, refletindo sua eficácia;
- **Tempo Médio para Correção de Bugs (MTTR):** Tempo necessário para corrigir problemas detectados na revisão. Um MTTR elevado pode indicar dificuldades na manutenção do código.

3.2 Importância da Análise de Métricas na Revisão

Monitorar essas métricas ajuda a garantir que o processo de revisão seja eficiente, equilibrando qualidade e velocidade. Revisões eficazes reduzem retrabalho, melhoram a colaboração e aumentam a confiabilidade do software. Além disso, é essencial evitar métricas que incentivem práticas prejudiciais, como priorizar velocidade em detrimento da qualidade.

4. Eficácia dos custos de revisões de software

No desenvolvimento de software, os custos com revisões e retrabalho são um grande desafio. No Brasil, estima-se que equipes de desenvolvimento gastem até 26% do tempo ajustando e corrigindo códigos antes do lançamento, o que representa um custo considerável para as empresas. Muitas vezes, os gestores de TI se concentram em cortar despesas com licenças de software e salários, mas ignoram o impacto das ineficiências no processo de desenvolvimento, que podem ser ainda mais prejudiciais. O retrabalho geralmente ocorre devido a requisitos mal definidos ou falhas na comunicação entre a equipe de desenvolvimento e os usuários. Estudos mostram que o custo de corrigir ou alterar o código pode ser até 2,5 vezes maior do que o custo de escrever o código original. Por isso, é fundamental que os requisitos sejam bem definidos desde o início e que técnicas como prototipagem sejam usadas para evitar mudanças tardias. Reduzir o retrabalho não só diminui os custos diretos, mas também melhora a produtividade da equipe, permitindo que o tempo economizado seja utilizado de forma mais eficiente em outras áreas do desenvolvimento. Com um bom gerenciamento de requisitos e melhores práticas, é possível otimizar o processo e entregar software de maior qualidade dentro dos prazos e orçamentos estabelecidos.

5. Técnicas de Revisão de Software

As revisões formais são realizadas normalmente por um pequeno grupo, incluindo um planejamento pré-revisão, a revisão em si e um pós-revisão. Assim ocasionando uma análise mais precisa de resultados/erros e etc.

Inspeção pertence à classe de revisões formais. É uma técnica mais precisa, com uso de checklists, tem o objetivo de descobrir defeitos e corrigi-los com um custo menor e eficiente no processo de desenvolvimento.

A revisão informal, no geral, é feita com um menor planejamento, sem uma agenda bem definida ou acompanhamento prévio.

Walkthrough ou Teste de mesa, é uma técnica com a ideia de analisar o passo a passo do artefato para identificar anomalias, melhorar o produto, treinar a equipe e analisar soluções.

5.1 Revisão Informal

As revisões informais ocorrem de maneira não-estruturada, não necessitam de um planejamento ou preparação, não possuem uma agenda definida. Normalmente, é feita de maneira espontânea e rápida. Esse tipo de revisão ocorre quando um desenvolvedor solicita a opinião de colegas sobre um código ou documento sem a necessidade de reuniões formais ou registros detalhados, por exemplo. Algumas formas de revisões informais são pair programming (programação em dupla), reuniões diárias ou semanais feitas de forma rápida sem a necessidade de reuniões formais ou registros bem detalhados.

5.2 Revisão Técnicas Formais

Diferente da Revisão Informal, as Revisões Técnicas Formais (FTR) seguem um formato totalmente estruturado garantindo a qualidade do software, com divisões de funções para participantes e diversas etapas como: Planejamento da Revisão, distribuição do software para Revisão, reunião de Revisão, relatório de Revisão e correção.
Com esse modelo de revisão baseado em reuniões, torna-se mais fácil identificar defeitos no software precocemente antes que se tornem um problema significativo, melhorando a qualidade e sua rastreabilidade e documentação do mesmo.

5.3 Relatório de Revisão e Manutenção de Registros

O Relatório de Revisão é um documento formal feito pelo moderador que contém todo o conteúdo resumido da reunião como: Informações sobre defeitos, soluções propostas e qualquer outra informação importante relacionada ao software. Esse relatório garante a documentação de todos as decisões e problemas identificados, ajudando também na sua rastreabilidade e conformidade.

A Manutenção de Registros é o processo de rastreabilidade do documento e informações das revisões, sendo guardados de forma organizada e acessível.

5.4 Conceitos e Técnicas de Revisão de Software

Ao falarmos sobre a revisão de software devemos abordar alguns métodos e práticas utilizados dentro do mercado, mas também explicar um pouco sobre o seu funcionamento e benefício, sendo ela, uma prática essencial para o ciclo de vida do desenvolvimento de sistemas. Permitindo a detecção precoce de defeitos e consequentemente a melhoria contínua de seus processos, garantindo a qualidade dos produtos desenvolvidos. Dentre as principais técnicas de revisão, destacam-se as diretrizes de revisão, as revisões por amostragem e as avaliações post-mortem.

5.5 Diretrizes de Revisão

As diretrizes de revisão referem-se a um conjunto de normas e boas práticas adotadas para avaliar os artefatos de software, como código-fonte, documentação técnica e modelos de design. Seu principal objetivo é garantir que o produto esteja alinhado com os requisitos estabelecidos e padrões de qualidade a fim de garantir a satisfação do cliente. Essas revisões eficazes seguem uma estrutura bem definida, realizada por profissionais qualificados, que utilizam listas de verificação e formalizam os resultados obtidos.

5.6 Revisões por Amostragem

As revisões por amostragem por outro lado selecionam partes representativas do software para fazer a validação em vez de analisar todo o sistema. Utilizada especialmente em cenários onde há restrições de tempo ou recursos, por permitir a identificação de padrões de erros

recorrentes e possíveis falhas críticas. Essa técnica possibilita uma avaliação significativa, servindo como um indicativo para determinar a necessidade de inspeções mais aprofundadas.

5.7 Avaliações Post-Mortem

As avaliações post-mortem, são conduzidas após a finalização do ciclo de desenvolvimento. Seu principal objetivo é analisar o desempenho do projeto como um todo, identificando pontos fortes e fracos, possibilitando oportunidades de melhoria para futuras iniciativas. São realizadas reuniões que permitem coletar o feedback da equipe, documentar lições aprendidas e estabelecer diretrizes para otimização dos processos.

6. Conclusão

A gestão e a qualidade de software desempenham um papel muito importante no desenvolvimento de produtos confiáveis e eficientes. A revisão de código, a análise de métricas e a aplicação de técnicas estruturadas de revisão são estratégias essenciais para minimizar defeitos e reduzir custos operacionais. Como pudemos identificar anteriormente, a detecção precoce de falhas e a eliminação de retrabalho não apenas garantem maior satisfação dos stakeholders, mas também otimizam o tempo e os recursos das equipes de desenvolvimento.

Além disso, a utilização de metodologias como inspeções formais, revisões por amostragem e avaliações post-mortem possibilita uma melhoria contínua dos processos, aumentando a eficiência e a competitividade das empresas no mercado. Dessa forma, investir em boas práticas de revisão de software não é apenas uma necessidade técnica, mas um fator determinante para o sucesso de projetos e organizações.

7. Referências

O Impacto da Qualidade de Software no Mercado – CBQS. Disponível em: <https://cbqs.com.br/2024/02/19/o-impacto-da-qualidade-de-software-no-mercado/>.

NUZZI, J. R. Quanto os erros em software podem custar para a sua empresa? Disponível em: <https://medium.com/@jonesroberto/quanto-os-erros-em-software-podem-custar-para-a-sua-empresa-409b77d08bd6>. Acesso em: 11 mar. 2025.

TORRES, E. Engenharia de Software e UML: Revisões, Auditoria e Inspeções Essenciais. Disponível em: <https://blog.faspec.edu.br/prevencao-de-defeitos-em-software/>. Acesso em: 11 mar. 2025.

Rational Software Architect RealTime Edition 9.5.0. Disponível em: <https://www.ibm.com/docs/pt-br/dmrt/9.5?topic=type-running-software-metrics>. Acesso em: 11 mar. 2025.

Disponível em: <https://youtu.be/nA1BVDd9GUE?si=AHs2APfFczaFJemY>. Acesso em: 11 mar. 2025.

BEATRIZ, A.; LOPES, A. REVISÃO DE SOFTWARE Baseado em materiais gentilmente cedidos por: Tayana Conte (UFAM) Guilherme Horta Travassos (COPPE-UFRJ). [s.l: s.n.]. Disponível em: <https://www.inf.ufpr.br/lmperes/2018_2/ci221/divulgacao/aula2_testes_revisao_de_sw.pdf>. Acesso em: 11 mar. 2025.

NEGRINI, C. Entendendo o Teste de Software por Amostragem. Disponível em: <https://www.devmedia.com.br/entendendo-o-teste-de-software-por-amostragem/31377>. Acesso em: 11 mar. 2025.

ATLASSIAN. Collaboration software for software, IT and business teams. Disponível em: <https://www-atlassian-com.translate.goog/?_x_tr_sl=en&_x_tr_tl=pt&_x_tr_hl=pt&_x_tr_pto=tc&_x_tr_hist=true#an-effective-incident-postmortem-plan>. Acesso em: 11 mar. 2025.

Medium. Disponível em: <https://medium.com/@celionormando/revis>. Acesso em: 11 mar. 2025.

TORRES, E. Engenharia De Software: Técnicas De Revisão. Disponível em: <https://blog.faspec.edu.br/software-tecnicas-de-revisao/>. Acesso em: 11 mar. 2025.

Revisão Técnica Formal (FTR). Disponível em: <https://julianakolb.wordpress.com/2014/01/20/revisao-tecnica-formal-ftr/>. Acesso em: 11 mar. 2025.

EQUIPE TOTVS. Tipos de relatório de manutenção e como elaborar o documento. Disponível em: <https://www.totvs.com/blog/gestao-para-assinatura-de-documentos/relatorio-de-manutencao/>. Acesso em: 11 mar. 2025.