Qualidade de Software

Four Bits

Manual Suplementar de Qualidade

Junho de **2025**

FOUR BITS

Um grupo de colegas de faculdade uniu seus conhecimentos e habilidades para desenvolver um sistema inovador, com um diferencial claro desde o início: a preocupação com a qualidade do software. Cientes da importância de entregar um produto funcional, confiável e sustentável, eles adotaram boas práticas de engenharia de software, testes contínuos e estratégias de garantia da qualidade durante todo o ciclo de desenvolvimento. Mais do que cumprir um requisito acadêmico, o projeto reflete o compromisso do grupo com a excelência técnica, a colaboração e a construção de soluções que realmente façam a diferença.

Manual técnico suplementar sobre Qualidade de Software

Desenvolvido por:

André Miquelino Campos
Diego Cunha
Lucas Eufrasio
Nicolas Robert de Oliveira Borges
Sandro Marcos Machiniski

Introdução

A crescente complexidade dos sistemas digitais e a demanda por soluções confiáveis e eficientes tornam a **Qualidade de Software** um fator essencial para o sucesso de qualquer projeto tecnológico. Este **manual técnico suplementar** foi elaborado com o propósito de oferecer uma base sólida de conceitos, práticas e ferramentas voltadas para a garantia da qualidade em todas as etapas do desenvolvimento de software.

Voltado especialmente para estudantes, profissionais em formação e equipes que buscam excelência em seus produtos, o material aborda temas como planejamento da qualidade, estratégias de teste, métricas, automação, além de boas práticas de documentação e melhoria contínua.

Sumário

1. Fundamentos e Filosofia da Qualidade

1.1.1. Conceitos de qualidade (Garvin, Deming, Juran)

No mundo da qualidade, excelência e gestão, existiram alguns pensadores que contribuíram significativamente para a área. David A. Garvin é um desses nomes. Este renomado acadêmico e pensador tem influenciado e moldado a forma como as organizações abordam a qualidade e a gestão.

David Garvin teve contribuições significativas para a área da gestão e da qualidade. O Modelo das Oito Dimensões da Qualidade é uma das contribuições mais notáveis de Garvin.

Ele delineou as características que definem a qualidade de um produto ou serviço: desempenho, características, confiabilidade, conformidade, durabilidade, serviço, estética e percepção.

Além disso, Garvin enfatizou a importância da melhoria contínua como uma abordagem fundamental para alcançar a excelência. Ele argumentou que as organizações devem se esforçar constantemente para aprimorar processos, produtos e serviços.

1.1.2. Conceitos de qualidade (Garvin, Deming, Juran)

De acordo com Deming, existia em gestão um mito de que qualidade e produtividade eram incompatíveis. Se se buscasse por qualidade, perdia-se em produtividade; e vice-versa.

Para Deming, esse é um **erro cujas consequências foram nefastas**: as organizações passaram a pensar qualidade e produtividade de maneira fragmentada, ter menos constância no planejamento de produtos e serviços, altíssima ênfase em ganhos de curto prazo, suposição de que precisariam de muita tecnologia para transformar o negócio. Estas por sua vez levaram a crenças como a de que qualidade significaria apenas seguir as especificações e de que a responsabilidade pela falta de qualidade seria sempre da equipe.

Deming desdobra o conceito de qualidade em 14 princípios.

1. Constância de Propósito: Empresas devem focar na melhoria contínua de produtos e serviços para se manterem competitivas a longo prazo.

2. Adote a Nova Filosofia: O mundo mudou e a gestão precisa evoluir, integrando qualidade com produtividade.

3. Elimine a Inspeção em Massa: A qualidade deve ser construída no processo, não checada no final.

4. Não Escolha Fornecedores só pelo Preço: Priorize parcerias duradouras baseadas em confiança e qualidade.

5. Melhoria Contínua: Otimize sempre os sistemas de produção e serviços desde o início do processo.

6. Treinamento Adequado: Capacite colaboradores para entenderem não só suas funções, mas o todo organizacional.

7. Liderança de Verdade: Supervisores devem apoiar e desenvolver as pessoas, não apenas cobrar números.

8. Elimine o Medo: Crie um ambiente seguro para que as pessoas se expressem e inovem.

9. Quebre Barreiras entre Departamentos: Promova o trabalho em equipe multidisciplinar.

10. Elimine Slogans e Metas Irrealistas: A pressão por resultados ignora que a maioria dos problemas vem do sistema, não dos indivíduos.

11. Elimine Cotas e Gestão por Números: Melhorar o sistema é mais eficaz do que impor metas numéricas.

12. Respeite o Orgulho Profissional: Dê ao trabalhador condições reais de se orgulhar do que faz.

13. Educação Contínua: Invista no desenvolvimento das pessoas em todos os níveis.

14. Envolvimento de Todos: A transformação da empresa precisa do esforço conjunto de todos os setores.

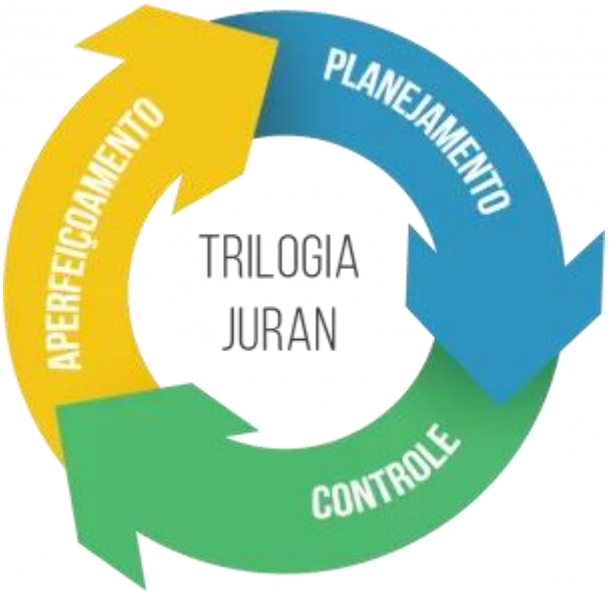
1.1.3. Conceitos de qualidade (Garvin, Deming, Juran)

Trilogia Juran

Uma de suas maiores contribuições, foi o desenvolvimento da **trilogia Juran para gerenciar a qualidade**, que também ficou conhecido como Juran Management System (JMS), ou Sistema de Gerenciamento Juran, em português.

A Trilogia é composta por:

- **Planejamento:** considerar a qualidade desejada e projetar meios para alcançá-la
- **Controle:** diagnosticar erros ou acertos no processo
- **Aperfeiçoamento:** propor patamares de qualidade cada vez mais altos



Ele também estabeleceu **7 princípios** para serem aderidos pelos gestores e líderes:

1. Estar disposto a entender as necessidades dos clientes e satisfazê-las;
2. Proporcionar alta qualidade de produtos e serviços, enquanto reduz custos;
3. Estar envolvido para identificar as necessidades dos clientes

4. Treinar e envolver a todos nos processos de gerenciamento para qualidade;
5. Agregar metas de qualidade ao planejamento de negócios;
6. Fornecer participações à força de trabalho;
7. Altos gerentes devem ter a iniciativa de realizar a gestão de qualidade.

Joseph foi mais uma figura importantíssima neste cenário, contribuindo não apenas onde trabalhou, mas também para muitas empresas que aderiram suas técnicas e padrões de organização.
Uma de suas principais frases sobre padronização: "*Não existe controle sem padronização".*

1.2. Custo da não qualidade

O custo da não qualidade é um conceito amplamente discutido no campo da gestão da qualidade. Refere-se aos custos associados à falta de qualidade em produtos ou serviços, incluindo os custos de retrabalho, reparo, devolução, perda de clientes e danos à reputação da empresa.
O custo da não qualidade é uma medida dos custos incorridos devido à falta de qualidade em produtos ou serviços. Esses custos podem ser divididos em duas categorias principais: custos internos e custos externos.

- Os custos internos são aqueles associados à detecção e correção de problemas de qualidade antes que eles cheguem aos clientes. Isso inclui o retrabalho, que é o processo de refazer um produto ou serviço para corrigir um defeito ou falha.
- Os custos externos são aqueles associados aos problemas de qualidade que afetam os clientes e a reputação da empresa. Isso inclui os custos de devolução de produtos defeituosos, reembolsos, perda de clientes e danos à imagem da marca.

O custo da não qualidade é um desafio significativo para as organizações, mas pode ser gerenciado de forma eficaz por meio de práticas de controle de qualidade robustas e uma cultura de qualidade. Ao investir em processos de controle de qualidade, envolver todos os funcionários, estabelecer parcerias com fornecedores confiáveis e monitorar continuamente a qualidade, as organizações podem reduzir os custos associados à não qualidade e garantir a satisfação do cliente.

1.3. Qualidade de produto vs. qualidade de processo

Qualidade de processos e qualidade de produtos são assuntos muito relacionados e por muitas vezes confundidos. Para iniciar, quero começar afirmando que as empresas são feitas de processos, certo? E esses processos servem para produzir alguma coisa, seja um produto ou algum tipo de serviço. Por este motivo, é essencial e importantíssimo que a qualidade tanto do processo quanto do produto tenha sua devida atenção.

No final das contas o resultado será a satisfação (ou insatisfação) do nosso cliente. Não importa qual negócio você tenha, tenho certeza de que a satisfação do seu cliente é importante para você. Essa satisfação está diretamente relacionada com a qualidade daquilo que você entrega.

Não existe pouca ou muita qualidade. A qualidade de alguma "coisa" é definida se essa "coisa" atende aos requisitos que deveriam ser atendidos (conformidade aos requisitos). Isso significa que ter qualidade é atender as especificações pré-estabelecidas.

A qualidade do produto ou serviço se refere basicamente ao nível em que este entregável atende aos requisitos previamente estabelecidos para ele.

Por exemplo, você estava em alguma pizzaria e solicitou uma pizza de atum e veio atum com queijo. Este produto (pizza) está **não conforme**. Especificado: atum > Realizado: atum com queijo > Resultado: produto não conforme!

1.4. Qualidade percebida pelo usuário final

Se a qualidade percebida fala sobre percepção do cliente, a qualidade real tem relação com a parte mais burocrática, ou seja, o desempenho, a performance, a estética e outros fatores. Para avaliar isso, alguns testes práticos e questões técnicas são observados. Além disso, existem formas

de dimensionar ambas as estratégias para que seja possível melhorar e alcançar a excelência do produto, a começar pela avaliação da satisfação do cliente.

Para começar, é preciso ter uma estratégia de marketing de relacionamento com o cliente, para mantê-lo envolvido com a marca. É necessário oferecer seu produto da melhor forma possível e claro, lembrando sempre do pós-venda. Aqui é importante manter contato ativo para saber se as expectativas foram atendidas e se ele ficou satisfeito com a solução.

Relacionamento pós-venda

Garantir a venda é só uma parte da estratégia, mas mantê-la mesmo após a venda também é importante. Fale com seu cliente, converse sobre o processo de compra, o relacionamento que ele teve com a sua empresa, como ele se sentiu, como está sendo a utilização do produto ou serviço, enfim, seja presente no pós-venda para que seu cliente se sinta valorizado pela empresa, aumentando a qualidade percebida.

1.5. Princípios ágeis e Lean aplicados à qualidade

As metodologias ágeis ajudam a levar eficiência aos negócios porque facilitam os processos, ajudam no alinhamento das equipes, na comunicação clara e permitem manter o foco em objetivos predeterminados. Como fogem totalmente da proposta de métodos tradicionais e burocráticos, representam uma grande revolução na gestão de empresas.

Para se ter uma ideia, uma consultoria mineira de soluções em nuvem obteve aumento de 23% em produtividade com o uso de métodos Kanban e Scrum, em relação a modelos comuns de gestão. Indicadores que refletem a satisfação dos clientes e o nível de ociosidade das equipes melhoraram, mesmo durante a fase mais aguda da pandemia.

Isso ocorre porque os sistemas de tarefas são muito mais dinâmicos, flexíveis e rápidos, o que se torna um excelente diferencial competitivo para a organização. Além disso, permitem uma rápida adaptação, tanto por parte dos envolvidos quanto em relação às constantes mudanças no mundo corporativo.

As empresas dispostas a adotarem a metodologia ágil em seus projetos favorecem o crescimento e a expansão dos negócios. A seguir, você descobre o que são, quais são suas principais vantagens e como aplicá-las na prática.

2. Modelos e Normas de Qualidade

2.1. ISO 9126 / ISO 25010 (qualidade do produto de software)

2.1.1. ISO 25010: O padrão mais recente

A ISO 25010 é a versão mais recente do padrão de modelo de qualidade de software, publicado em 2011. Ele substitui a ISO 9126, que foi emitida pela primeira vez em 1991 e revisada em 2001. A ISO 25010 expande e reorganiza as características e subcaracterísticas da qualidade de software e introduz duas novas perspectivas: qualidade no uso e qualidade no contexto. Qualidade em uso refere-se ao grau em que um produto de software satisfaz as necessidades e objetivos de seus usuários em uma situação específica, enquanto qualidade em contexto refere-se ao grau em que um produto de software se adapta a diferentes ambientes e condições.

2.1.2. ISO 9126: O padrão mais antigo

A ISO 9126 é a antecessora da ISO 25010 e define seis características principais da qualidade de software: funcionalidade, confiabilidade, usabilidade, eficiência, manutenibilidade e portabilidade. Cada característica tem uma série de subcaracterísticas que especificam ainda mais os atributos de qualidade de software. A ISO 9126 também fornece uma estrutura para avaliar a qualidade de software com base em métricas internas, externas e de qualidade em uso. As métricas internas medem a qualidade do código e do design do software, as métricas externas medem a qualidade do

comportamento e do desempenho do software e as métricas de qualidade no uso medem a qualidade do software do ponto de vista do usuário.

2.2. ISO 12207 (processos de ciclo de vida)

O escopo da Norma ISO/IEC 12207 abrange todo o ciclo de vida de software, desde a concepção inicial até a descontinuidade do software, e por todos os envolvidos com produção, manutenção e operação do software. A norma pode aplicada para toda empresa desenvolvedora de software, mas existem casos de aplicação em projetos específicos por imposição contratual ou nas fases iniciais de implantação.

Os processos da Norma ISO/IEC 12207 são agrupados de acordo com o seu objetivo principal no ciclo de vida de software. Estes agrupamentos resultam em três classes de processos: Processos Fundamentais, Processos de Apoio e Processos Organizacionais. Segundo Arruda (2006), a classe dos Processos Fundamentais são basicamente todas as atividades que a empresa executa nos serviços de desenvolvimento, manutenção ou operação de software. Esses processos comandam a execução de todos os outros processos. Os cinco processos fundamentais de ciclo de vida são:

a) Aquisição;

b) Fornecimento;

c) Desenvolvimento;

d) Operação;

e) Manutenção.

2.3. ISO 29119 (testes de software)

A série ISO/IEC/IEEE 29119 é composta por cinco partes que visam padronizar processos, documentação e técnicas de teste de software. Os padrões promovem uma abordagem estruturada e baseada em risco, aplicável a qualquer modelo de ciclo de vida de desenvolvimento (SDLC).

- 29119-1 (Conceitos e Definições):
  Estabelece a base conceitual dos padrões, definindo termos, processos e formas de aplicação. Serve como referência para a adoção das demais partes da série.
- 29119-2 (Modelo de Processo de Teste):
  Define um modelo genérico para governar, gerenciar e implementar testes. Prioriza testes com base em riscos, otimizando esforços para garantir os atributos mais críticos de qualidade.
- 29119-3 (Documentação de Teste):
  Apresenta modelos padronizados para documentação ao longo de todo o ciclo de vida do teste. Substitui o antigo padrão IEEE 829 e permite adaptações conforme as necessidades de cada organização.
- 29119-4 (Técnicas de Design de Teste):
  Foca em métodos de criação de casos de teste. Deriva do padrão BS-7925 e oferece diretrizes para diferentes abordagens de design, dependendo do contexto e do modelo SDLC.

- 29119-5 (Testes por Palavras-chave):
  Introduz diretrizes para testes baseados em palavras-chave. Apesar de ainda não publicado à época, pretende padronizar esse tipo específico de teste funcional.

Complemento:

- ISO/IEC 33063:
  Um modelo de avaliação de processo baseado no 29119-2. Fornece indicadores para avaliar a maturidade e efetividade dos processos de teste implementados.

2.4. CMMI (níveis de maturidade)

Os níveis de maturidade do CMMI e como eles ajudam a evoluir os processos organizacionais:

Nível 1 – Inicial

Processos são improvisados e reativos. Há atrasos, desperdícios e falta de planejamento. O sucesso depende de esforços individuais, sem padronização.

Nível 2 – Gerenciado

Projetos passam a ser planejados e controlados. Há gestão de prazos e custos, mas ainda com perdas e ineficiências. A cultura de melhoria começa a se formar.

Nível 3 – Definido

Processos são padronizados e documentados em toda a organização. A cultura organizacional é fortalecida e os colaboradores atuam de forma alinhada.

Nível 4 – Gerenciado Quantitativamente

A empresa utiliza dados e análises estatísticas para tomar decisões. Os processos são estáveis, previsíveis e orientados por métricas.

Nível 5 – Otimização

Foco total em melhoria contínua e inovação. Os processos são flexíveis, adaptáveis e constantemente aprimorados para aumentar a competitividade.

Esses níveis formam uma escada evolutiva que ajuda as empresas a se tornarem mais eficientes, confiáveis e inovadoras.

2.5. MPS.BR (modelo brasileiro)

O MPSBR, Melhoria do Processo de Software Brasileiro, é um programa da Softex lançado em dezembro de 2003 como uma iniciativa brasileira para promover da qualidade do desenvolvimento de software, de serviços e gestão de pessoas pela melhoria de processos, aumentando a produtividade das equipes, diminuindo o retrabalho e fomentando a competitividade da empresa. O programa tem por objetivo melhorar a capacidade de desenvolvimento de software, serviços e as práticas de gestão de pessoas na indústria de TIC.

2.6. SPICE / ISO 15504 (avaliação de processos)

A ISO/IEC 15504 é uma norma internacional para a certificação dos processos de negócio através de avaliações correspondentes. O nome "SPICE" (Software Process Improvement and Capability Determination) é o nome do projeto a partir do período de desenvolvimento da norma. Ele

foi desenvolvido em várias etapas a partir de 1993 e recebeu o status de norma em 2006 sob a designação ISO/IEC 15504.

O SPICE considera, por um lado, a melhoria dos processos dentro da própria organização e, por outro, a determinação da capacidade do processo. Portanto, a norma contém requisitos para modelos de referência de processos (PRM) para descrição de processos, bem como modelos de avaliação de processos (PAM) com critérios e métodos para a sua avaliação.

A dimensão de capacidade ou maturidade, que divide um processo de desenvolvimento em seis níveis de "incompleto" a "otimizado", é considerada como um componente particularmente útil para a prática. Desde 2015, a ISO têm transferido sucessivamente o conjunto de regras em dez partes para a nova série de normas ISO/IEC 330xx.

2.7. ITIL (qualidade de serviços de TI)

A ITIL é um conjunto de ensinamentos e boas práticas com a finalidade de identificar, planejar e entregar um bom suporte de demandas de TI. A metodologia surgiu nos anos 80, e vem sendo atualizada constantemente.

Hoje, podemos afirmar que ela é a principal referência para gerenciamento de TI, e discorre sobre todos os processos e atividades, indo além da gestão de infraestrutura à que se propôs originalmente.

A ITIL (Information Technology Infrastructure Library) surgiu no Reino Unido, desenvolvida pela antiga CCTA (atualmente Cabinet Office), com o objetivo de documentar e padronizar práticas de TI utilizadas por empresas que prestavam serviços ao governo. Inicialmente um guia de recomendações, a ITIL evoluiu ao incorporar experiências de outras organizações, consolidando-se como um modelo de boas práticas em gestão de serviços de TI.

Nos anos 1990, a ITIL ganhou força na Europa e, posteriormente, se espalhou globalmente. Atualmente, é mantida pela Axelos, uma joint venture entre o governo britânico e a empresa Capita plc.

3. Qualidade em Engenharia de Requisitos e Análise

3.1. Verificação e validação de requisitos

No ambiente de desenvolvimento acelerado de hoje, garantir o sucesso de um projeto depende fortemente de processos robustos de verificação e validação de requisitos. Essas duas etapas cruciais do processo de verificação e validação ajudam a identificar erros, alinhar as expectativas das partes interessadas e garantir a conformidade com os padrões do setor. As organizações podem reduzir significativamente os riscos do projeto, otimizar o desenvolvimento e entregar resultados de alta qualidade validando se os requisitos atendem às necessidades do usuário e verificando sua precisão e integridade.

Este guia explora a importância da verificação de requisitos, melhores práticas para validação de requisitos e as ferramentas e soluções disponíveis para simplificar e aprimorar esses processos. Seja você um gerente de projeto, analista de negócios ou desenvolvedor, dominar essas práticas é essencial para criar produtos que atendam às expectativas funcionais e das partes interessadas.

3.2. Análise de ambiguidade e inconsistência

Identifique as fontes de ambiguidade:
Use ferramentas como análise de partes interessadas ou SWOT para descobrir onde há incerteza no projeto e como ela afeta os resultados.

Busque esclarecimentos e feedbacks:
Comunique-se com as partes envolvidas para alinhar expectativas, reduzir mal-entendidos e validar o progresso.

Abrace o aprendizado e a experimentação:
Use protótipos, testes e pilotos para aprender com tentativas e falhas, explorando soluções com abertura para ajustes.

Colabore e aproveite a diversidade:
Incentive diferentes perspectivas por meio de fóruns, enquetes e plataformas colaborativas. A troca entre diferentes visões enriquece o processo.

Seja flexível e resiliente:
Adote estratégias como planos de contingência e práticas de autocuidado para enfrentar mudanças e imprevistos com equilíbrio.

Revise e reflita:
Analise resultados, processos e lições aprendidas. Isso permite identificar melhorias contínuas e reforçar boas práticas.

Considere outros pontos relevantes:
Espaço aberto para exemplos, histórias e aprendizados que podem ampliar o entendimento e enriquecer a abordagem.

3.3. Técnicas para melhorar a qualidade de especificações

Gestão da qualidade é um conjunto de práticas que ajudam a garantir que os produtos ou serviços oferecidos pela empresa atendam ou excedam as expectativas dos clientes.

Com a melhoria contínua da empresa, a retenção de clientes tende a aumentar. Por isso, a implementação eficaz da gestão da qualidade pode resultar em muitos benefícios, como:

- Maior satisfação do cliente;
- Redução de custos;
- Melhoria da reputação da empresa;
- Conformidade com normas regulamentadoras e padrões de qualidade;
- Entre outros.

A gestão da qualidade envolve não apenas o produto ou serviço em si, mas toda a experiência do cliente, incluindo etapas como atendimento, facilidade ao resolver algum problema e outras abordagens que priorizem a experiência do cliente.

3.4. Rastreabilidade e cobertura de requisitos
A Requirements Traceability Matrix (RTM) é uma ferramenta crítica no ciclo de vida dos requisitos, projetada para garantir o rastreamento e alinhamento abrangentes dos requisitos com artefatos

relacionados, como elementos de design, casos de teste e entregas. Ao fornecer uma estrutura clara para rastrear requisitos, a RTM permite que as equipes mantenham a visibilidade, gerenciem dependências e validem os resultados do projeto em relação às necessidades das partes interessadas.

A importância do RTM no desenvolvimento de software e sistemas é profunda. Ele não apenas aumenta a responsabilidade e simplifica a conformidade, mas também garante que todos os requisitos sejam atendidos efetivamente durante todo o processo de desenvolvimento. Em sua essência, o RTM é construído sobre a base da rastreabilidade de requisitos, um processo que garante que cada requisito esteja conectado à sua fonte e vinculado às fases correspondentes do ciclo de vida do projeto.

4. Testes em Profundidade

4.1. Teste baseado em risco

Os riscos estão presentes em diversas áreas de conhecimento, porém eles têm um significado distinto em cada contexto. A definição generalizada do termo, segundo a Wikipedia:

É a possibilidade de algo ruim acontecer. Envolve incertezas/implicações sobre o efeito de uma atividade com relação a algo que os humanos valorizam, geralmente focando nas consequências negativas e indesejáveis.

No contexto de engenharia de software, os riscos podem estar relacionados diretamente ao gerenciamento do projeto como um todo ou voltados estritamente para o uso da aplicação, focando nas possíveis experiências negativas que serão percebidas pelos usuários do produto. É aí que entra a abordagem de testes baseada em riscos!

Segundo um artigo da ReQtest, essa técnica checa a probabilidade de ocorrência de falhas na aplicação, utilizando casos de teste previamente criados para prever o impacto do que foi desenvolvido nos negócios do cliente. Além disso, partes críticas da aplicação são priorizadas, onde a probabilidade de ocorrer falhas é maior e onde também deve ser investido esforço na correção de bugs críticos.

4.2. Teste baseado em modelo (MBT)

Model-Based Testing (MBT) é uma abordagem poderosa de teste de software que usa modelos para representar o comportamento desejado de um sistema em teste. Ao automatizar a geração e execução de casos de teste, o MBT garante maior cobertura, eficiência e precisão de teste, tornando-se uma estratégia essencial para o desenvolvimento de software moderno. Ao contrário dos métodos de teste tradicionais, o MBT depende de modelos visuais ou matemáticos, permitindo que os testadores validem a funcionalidade do sistema de forma mais eficaz.
Model-Based Testing (MBT) é uma abordagem de teste de software que usa modelos para definir o comportamento esperado de um sistema. Esses modelos agem como blueprints, descrevendo entradas, processos e saídas do sistema, e são usados para gerar automaticamente casos de teste. Ao focar na lógica e nos requisitos do sistema, o MBT garante uma cobertura de teste abrangente e reduz a probabilidade de erro humano no design do teste.

4.3. Testes orientados por comportamento (BDD)

Esta metodologia é útil em projetos de software ágeis, que são construídos em várias iterações e estão sofrendo alterações ao longo do seu ciclo de vida. Quanto maior o projeto, mais difícil será a comunicação. Entretanto, BDD propõe uma forma eficaz de resolver estes problemas.

Quando se estuda Engenharia de Software, aprende-se que todo sistema tem um tempo de vida útil, e que uma série de fatores podem contribuir para aumentar ou diminuir este tempo: arquitetura, modelagem, tecnologia utilizada, aceitação do mercado, etc. No entanto, nenhuma empresa desenvolve um sistema pensando no dia em que ele deixará de atender as necessidades do seu cliente, apesar desta ser uma verdade certa.

Neste ciclo, o sistema passa a maior parte do tempo sofrendo alterações. E estas manutenções, sejam elas corretivas ou não, podem melhorar ou piorar o sistema, dependendo da forma que forem implementadas. Pensando neste e em outros problemas, Kent Beck apresentou ao mundo em 2003, através do seu livro “Test-Driven Development”, uma técnica para criar sistemas baseados em testes que visa garantir a qualidade e a funcionalidade do software durante este ciclo.

Embora TDD seja uma técnica testada e aprovada por grandes desenvolvedores ágeis, muitas equipes de desenvolvimento ainda caem em algumas armadilhas e mal-entendidos do tipo: por onde começar, o que testar e o que não testar. Isto sem falar que quem escreve os testes são os desenvolvedores, mas quando a equipe de qualidade vai testar, ela se preocupa com o comportamento do sistema e não com os testes unitários. Desta forma, não há comunicação eficiente entre as duas equipes no nível de código.

Para tornar a aplicação de TDD mais simples e ajudar as equipes de desenvolvimento a resolverem as questões mencionadas acima, **surgiu o Behaviour Driven Development (BDD)**. Neste artigo, veremos como aplicar esta técnica de forma eficiente, utilizando os frameworks mais conhecidos da comunidade Java.

4.4. Testes exploratórios

A atividade de teste de software é um dos meios utilizados para garantir a qualidade do sistema, evitando assim surpresas desagradáveis em relação ao resultado final da aplicação. O teste parece ser uma coisa simples e empírica, mas na verdade não é. E as falhas, muitas vezes, acontecem por não darmos a devida importância a esse fato. Neste contexto, abordamos neste artigo a definição e utilização de testes exploratórios, mostrando suas vantagens e desvantagens.

Testes exploratórios são extremamente úteis para encontrar bugs rapidamente, e quando o projeto não tem uma documentação bem definida ou a mesma encontra-se desatualizada. O que precisamos, hoje, é mostrar a importância desses testes, para garantia de qualidade dos sistemas. E isso só é possível introduzindo-se métodos para auxiliar no planejamento e especificação dos mesmos, pois o teste é um item fundamental para o desenvolvimento de sistemas.

A atividade de teste muitas vezes é vista como uma fase de crítica aos analistas e programadores, fazendo com que estes se sintam constrangidos. Mas, se os erros existem, o fato de não se fazer os devidos testes não quer dizer que eles irão desaparecer e, muitas vezes, se esses erros são detectados tardiamente podem causar prejuízo e danos incontroláveis aos sistemas. Dentre os vários tipos de testes existentes, neste artigo vamos definir, contextualizar, citar as vantagens e desvantagens e mostrar a aplicabilidade dos testes exploratórios.

4.5. Testes em aplicações móveis, web, IoT, embarcados

Dispositivos vestíveis, edificações, sistemas de energia, controles industriais inteligentes, RFID/NFC, etc. Tudo isso faz parte do mercado em expansão de Internet das Coisas (IoT). Cada uma destas tecnologias emprega padrões de conectividade wireless, como Bluetooth®, Wi-Fi, ZigBee, Z-wave, Wi-SUN, WirelessHART, ASK, FSK e GFSK.

Realizar testes de RF nestes dispositivos é essencial para o sucesso do mercado de IoT. Por este motivo preparamos uma série de documentos técnicos que foca em testes de RF para dispositivos de IoT. Tudo isto preparado pela empresa que é a líder do mercado de teste e medição eletrônica no mundo.

Cada um deles descreve um típico caso de teste de RF para um dispositivo de IoT e inclui exemplos de teste passo a passo sobre os seguintes tópicos:

- A Internet das Coisas: Possibilitando Tecnologias e Soluções para Projetos e Testes;
- Teste da vida útil da bateria e seus desafios com sensores wireless para Internet das Coisas;
- Solução flexível para teste de RF em transceptor Zigbee de 2,4 GHz;
- Solução flexível para teste de RF em dispositivos IoT com análise de sinais ASK/FSK;
- Caracterização de etiquetas RFID de baixa frequência.

5. Planejamento Estratégico da Qualidade

5.1. Plano de garantia da qualidade

Não existe um único processo de garantia da qualidade. Ela se apresenta como um conjunto de métodos que, aplicados dentro do processo, vão dar segurança de que ele atende os requisitos aplicáveis.

Esses método incluem:

- Mapas de processo
- Procedimentos e instruções de trabalho
- Programas de inspeção e testes
- Coleta e manutenção de evidências
- Programas de calibrações e manutenções
- Programas de auditoria
- Gestão de riscos etc.

Por ter um caráter sistemático, envolvendo diferentes fases da etapa de produção, a estruturação da garantia da qualidade envolve um intenso planejamento. Em muitos aspectos, ele converge com a estruturação do próprio SGQ. Assim como o próprio SGQ, ela deve estar focada na melhoria contínua.

5.2. Matriz de responsabilidade (RACI)

Uma matriz RACI é um gráfico que define e documenta a propriedade e a responsabilidade em cada estágio do seu projeto. Dentro da matriz, tarefas, marcos e decisões no projeto são mapeados por completo.

O termo RACI é uma sigla do inglês que designa quatro funções principais dentro de um projeto:

Responsible (encarregado): pessoa ou pessoas incumbidas de concluir a tarefa ou função atribuída. São encarregadas de concluir o trabalho ou tomar uma decisão. São as pessoas "que executam".

Accountable (responsável): a pessoa responsável pela conclusão bem-sucedida do trabalho (ou seja, o “dono” do trabalho). Essa pessoa aprova o trabalho depois de concluído. A parte encarregada responde a essa pessoa. Essa função geralmente é preenchida pelo patrocinador do projeto.

Consulted (consultado): pessoas que fornecem contribuições e informações necessárias sobre o projeto e as tarefas. Geralmente são especialistas no assunto e desempenham um papel ativo no processo.

Informed (informado): pessoas ou partes envolvidas que precisam ser mantidas informadas com relatórios de progresso e atualizações. Não contribuem diretamente para as decisões, mas precisam estar cientes delas.

5.3. Definição de critérios de aceite

Os critérios de aceite são estabelecidos e cada critério apresenta requisitos a serem cumpridos junto com as regras de negócio. Feito isso, ao final, na Sprint Review (reunião de revisão), a equipe apresentará ao Product Owner a entrega do desenvolvimento de forma que atenda a estória do usuário escrita e aos Critérios de Aceite e as Regras de Negócio.

Sempre que a User Story apresentada cumprir com os critérios de aceite, e atender as regras de negócio, receberá validação quanto a sua entrega.

5.4. Gerenciamento de configuração e controle de mudanças

As mudanças são constantes e inevitáveis no que tange o desenvolvimento de software. Isso ocorre porque o entendimento dos usuários mudam conforme as suas necessidades, o ambiente muda constantemente, a legislação está em constante mudança e os requisitos também sempre mudam. Com tantas mudanças necessitamos de alguma forma de gerenciamentos para que o desenvolvimento não siga em direção ao caos.

Dessa forma, surgiu a Gerência de Configuração de Software (GCS), ou em inglês Software Configuration Management (SCM), que é um conjunto de atividades de apoio que permitem controlar as mudanças que ocorrem no desenvolvimento de software, mantendo a estabilidade na evolução do projeto.

Entre outras coisas a Gerência de Configuração de Software responde a algumas questões como: Quais mudanças aconteceram no sistema? Por que essas mudanças aconteceram? O sistema continua íntegro mesmo após as mudanças?

Para entendermos um pouco mais o que vem a ser a Gerência de Configuração de Software é interessante definirmos o que vem a ser configuração. Configuração de um sistema é uma coleção de versões específicas de itens de configuração como hardware ou software que são combinados de acordo com procedimentos específicos de construção para servir a uma finalidade particular. A Gerência de Configuração de Software por sua vez é uma disciplina que identifica a configuração de um sistema em diferentes pontos no tempo com a finalidade de controlar sistematicamente as mudanças realizadas, mantendo a integridade e rastreabilidade da configuração através do ciclo de vida do sistema.

Alguns dos produtos que podemos colocar sobre o gerenciamento de configuração incluem: Hardware e equipamentos, desenhos ou modelos, especificação de produtos, ferramentas, códigos e bibliotecas, compiladores, ferramentas de testes e scripts de testes, log de instalação, publicações técnicas de produtos, planos, histórias de usuários, backlog das iterações, descrições dos processos, requisitos, documentação de arquitetura, processos, etc.

Portanto, a configuração é o estado do conjunto de itens que formam o sistema em um determinado momento e a Gerência de Configuração de Software é quem controla a evolução dessas configurações durante o ciclo de vida do projeto.

5.5. Controle de versões e auditoria de builds

O gerenciamento de builds e versões, também conhecido como gerenciamento de builds e lançamentos ou gerenciamento de builds, é o processo de gerenciar a criação, os testes e a distribuição de builds e lançamentos de versões de um software. Ele engloba coordenar as atividades envolvidas na criação de um build, no teste desse build e no seu lançamento para os usuários.

O gerenciamento de builds e versões é uma parte importante do processo de desenvolvimento de software porque ajuda a garantir que os builds e as versões lançadas sejam criados e distribuídos de maneira controlada e consistente. Ele também ajuda a garantir que os builds e as versões lançadas sejam de alta qualidade e estejam prontos para uso pelos usuários.

Em geral, o gerenciamento de builds e versões envolve uma série de fases, entre elas:

1. Compilação e construção do software: inclui a compilação do código-fonte em um formato executável, bem como realizar outras atividades, como rodar testes, criar a documentação e empacotar o software para distribuição.
2. Teste do build: envolve verificar se o build é de qualidade suficiente e se atende às especificações exigidas. Testes podem incluir testes manuais, testes automatizados ou ambos.
3. Gerenciamento de dependências: inclui garantir que todas as dependências necessárias sejam incluídas no build e estejam configuradas corretamente.
4. Gerenciamento de versões: inclui acompanhar as versões do software que foram lançadas, bem como manter um histórico de alterações e correções de bugs.
5. Distribuir o lançamento: envolve disponibilizar o lançamento aos usuários por meio de uma loja de aplicativos, site ou outro canal de distribuição.

6. Fatores Humanos e Organizacionais

6.1. Cultura de qualidade dentro da equipe

A cultura da qualidade está presente quando há o comprometimento de toda a equipe – não só da direção e gestores – com os processos e a implementação correta. Ou seja, essa prática exige que

toda a equipe compreenda o real significado da gestão de qualidade, envolvendo não só a eficiência operacional, mas também a manutenção de um clima corporativo agradável.

Quando a cultura da qualidade faz parte do DNA de uma empresa, existe a percepção da importância em atender às necessidades de clientes somado ao mérito do envolvimento de cada colaborador para o alcance das metas estabelecidas.

Um novo olhar, mais atento, para os processos internos de qualidade traz como benefício a melhoria na performance da empresa, pois todos os envolvidos desempenham seu trabalho e funções em acordo com um único propósito. Para te ajudar nessa tarefa, listamos algumas práticas que vão manter seu grupo sempre alinhado com a missão e valores do negócio

6.2. Comunicação entre áreas (Dev, QA, Produto)

A importância de uma boa comunicação em QA é um pilar essencial para o sucesso em projetos de desenvolvimento de software, ainda mais quando nos referimos à Qualidade. Não basta apenas transmitir informações; é necessário identificar e solucionar desafios de maneira eficiente, mas também contribuir para um ambiente colaborativo.

Ou seja, manter a boa comunicação ajuda a impulsionar a qualidade, satisfazer os clientes e alcançar o sucesso no desenvolvimento do projeto.

6.3. Papel do QA em times ágeis

Você possivelmente já deve ter ouvido falar em Waterfall e desenvolvimento ágil no seu ambiente de trabalho. Mas o que realmente isso significa? Para entendermos o contexto e nos aprofundarmos um pouco mais nos testes, precisamos conhecer um pouco dos dois, pois a diferença da aplicação de testes se fundamenta principalmente na abordagem dos times. Vamos lá?

O modelo Waterfall ou modelo cascata é tradicionalmente usado, e dentro dele os projetos são construídos em etapas.Nesse formato, existe apenas um caminho a ser seguido e a finalização do projeto nem sempre resulta em clientes satisfeitos. Em relação aos testes, o primeiro passo é construir o projeto e apenas depois revisar para encontrar possíveis bugs e problemas. Depois da revisão, o produto volta para a etapa de desenvolvimento para realizar os fixes. Os requisitos definidos inicialmente são contemplados, mas o feedback do cliente tende a demorar..

Então, o que fazer se o produto não estiver alinhado com as expectativas do usuário final ou os ganhos do cliente simplesmente não existirem por conta de um bug que só foi identificado após a finalização das etapas?

Imaginem então o time de Quality Assurance nesse contexto. Quando um produto chega ao cliente ou entra em produção, o esperado é que seja consumido em escala. É comum pensar que os testes farão mais sentido quando o produto estiver finalizado, mas você já se perguntou se esse procedimento, realizado apenas no final, garante a qualidade necessária para o projeto?

Podemos resumir o questionamento com a pergunta: nesse formato, é mais fácil um tester Quality Assurance encontrar um bug ou 50/100 pessoas encontrarem vários bugs durante o consumo do produto final?

6.4. Gestão do conhecimento e documentação viva

Segundo Takeuchi, esta é a primeira fase de transmissão de conhecimento, que é a Socialização (Transmissão do conhecimento de Tácito para Tácito), onde é transmitido entre pessoas, e justamente por isso, pode se perder ou sofrer alterações facilmente. Semelhante em como ocorre a brincadeira do “telefone sem fio”, uma informação sai da fonte de um jeito e passa por tantas pessoas que chega no final totalmente distorcida.

Com isso, surge também outro problema, pois colaboradores com conhecimentos avançados em determinados processos podem sair de férias, ficar doentes ou simplesmente não fazer mais parte da empresa, fazendo com que a organização, além de prejudicada, perca rendimento, visto que as tarefas realizadas por eles não são documentadas, e serão realizadas por outros colaboradores de formas divergentes, sendo talvez mais trabalhosa pela falta de experiência.

E se houvesse registros de como funciona cada tarefa? Assim inicia-se a segunda fase de transmissão do conhecimento, a Externalização (Transmissão do conhecimento de Tácito para Explícito), onde é registrado e formalizado em documento todo conhecimento importante para a realização das atividades que os colaboradores possuem e compartilham. Esse desafio ocorre frequentemente nas organizações, principalmente quando um novo colaborador precisa utilizar um sistema. É necessário ao colaborador conhecê-lo, saber acessá-lo, usar as funcionalidades, entender as regras de negócio, dentre outras funções, ele necessita aprender a usá-lo para executar seu trabalho. É facilitador quando um funcionário novo, ou até mesmo àqueles que trabalham a anos com o sistema, podem acessar um documento e nele tirar suas dúvidas sem precisar perguntar a alguém.

Nessa etapa, é importante que as empresas criem um registro textual do seu software, ou seja, uma Documentação. Nela, devem constar todas as informações necessárias para compreensão e utilização do software, exercendo um papel de ferramenta de auxílio na busca por questionamentos dos usuários durante seu uso.

6.5. Treinamento contínuo de times de desenvolvimento/teste

Para realizar melhores entregas, os desenvolvedores de software têm utilizado cada vez mais práticas ágeis e automatizadas. Com o teste contínuo, a validação do desempenho do software é constante e não apenas no final do ciclo. Dessa forma, são entregues feedbacks rápidos que podem evitar problemas futuros.

A identificação da causa raiz da falha pode ser mais difícil conforme mudanças são realizadas no software, onde qualquer reparo pode provocar uma reação em cadeia de ciclos de QA. Caso o desempenho não esteja atualizado dessa maneira, a tendência é que o todo o cronograma de lançamento do software seja adiado ou até mesmo cancelado.

Diante disso, adotar a prática de testes contínuos tem como objetivo principal verificar possíveis falhas no momento em que aconteçam, garantindo que a qualidade do software seja verificada em todas as etapas de produção.

7. Qualidade em Ambientes Ágeis e DevOps

7.1. QA contínua (Continuous Quality)

A qualidade contínua (CQ) é uma prática de engenharia que mede, monitora, melhora e depura a qualidade do software em todo o ciclo de vida de desenvolvimento de software (SDLC). Essa prática tem como objetivo avaliar a qualidade do software a cada alteração de código feita no sistema. O CQ é um componente importante da integração contínua (CI) e da entrega contínua (CD). O CQ é normalmente implementado antes do CD. Isso ocorre porque o CQ ajuda a garantir que o software seja de alta qualidade antes de ser implantado na produção. Como indústria, adotamos a implantação contínua. Isso significa que o software é implantado continuamente na produção, geralmente várias vezes ao dia. A necessidade de implementar o CQ é imperativa para garantir que o software seja de alta qualidade e que atenda às necessidades dos usuários

7.2. Shift-left testing e shift-right

É fundamental realizar testes em todo o ciclo de vida de desenvolvimento do software para atender às expectativas do usuário, aos requisitos de funcionalidade e às medidas de segurança. Realizar shift left é incorporar testes de segurança logo no início do processo para encontrar vulnerabilidades e corrigir defeitos o mais rápido possível no desenvolvimento. Realizar shift right é monitorar o comportamento, uso, desempenho e métricas de segurança do usuário no estágio de produção para verificar a operabilidade do software.

Os dois recursos têm como objetivo avaliar e garantir a qualidade e o desempenho de novas soluções e funcionalidades em todo o processo de DevOps e ciclo de vida de desenvolvimento do software (SDLC), concentrando-se em métodos de testes contínuos. A lógica por trás dos princípios de "shift left" e "shift right" em uma prática ágil é "falha pequena, recuperação rápida". Ela tem como objetivo detectar riscos antes que se transformem em problemas.

7.3. Integração de qualidade em pipelines DevOps

O conceito de DevOps tem evoluído como uma estratégia essencial para melhorar a eficiência operacional nas empresas de tecnologia. A adoção de pipelines DevOps visa acelerar a entrega de software, reduzir falhas, e melhorar a colaboração entre as equipes de desenvolvimento e operações.

Segundo o State of DevOps Report 2023, equipes que implementam práticas robustas de Continuous Integration e Continuous Delivery (CI/CD) conseguem aumentar sua performance operacional em até 50%. Esses times se beneficiam de ciclos de desenvolvimento mais rápidos e de uma maior capacidade de recuperação em caso de falhas, tornando o processo de lançamento de produtos mais ágil e eficiente.

Além disso, o uso de ferramentas de automação e orquestração, como o Kubernetes, tem crescido exponencialmente. Em 2023, quase metade das empresas implementaram Kubernetes em seus pipelines DevOps.

Outro aspecto importante é a integração de DevSecOps, que traz a segurança para o centro das operações DevOps, aumentando a governança e a proteção de dados desde as primeiras etapas do desenvolvimento. Isso ajuda a garantir a conformidade e reduz riscos de segurança sem comprometer a agilidade operacional.

Esse cenário mostra que a implementação de pipelines DevOps vai além da simples automação, envolvendo práticas de monitoramento, segurança e orquestração que impulsionam a eficiência operacional e a capacidade de inovação das empresas.

7.4. Feature flags, canary releases e rollback controlado

Dia após dia as empresas vem buscando formas de entregar mais funcionalidades, para o público certo, no tempo certo e o ponto mais crítico - com excelente desempenho e baixo volume de erros.

Diante deste cenário, os responsáveis pelos deployments e releases tem buscado estratégias avançadas de entrega que permitam entregar aplicativos e funcionalidades de forma mais rápida e confiável. Hoje falaremos sobre duas estratégias complementares: Canary Release e Feature Flags.

A estratégia de entrega Canary ajuda a alterar artefatos em ambiente produtivo, garantindo que a nova versão de sua aplicação esteja rodando com segurança, controlando o tráfego, e fazendo o que precisa fazer usando métricas de monitoração em conjunto a técnicas de machine learn e inteligência artificial para comparar as 2 versões em execução, até a virada total para a nova versão.

A estratégia Canary é ótima para backend e mudanças focadas em infraestrutura - bem como equalizar as funcionalidades em produção que você deseja começar a testar que cada deploy contém. O objetivo de uma estratégia de entrega canary é sempre fazer com que a nova versão chegue a 100% e, idealmente, fazê-lo o mais rápido possível.

O Canary deployment, entretanto, não é comumente utilizado para testes de novos recursos, execução de programas beta ou alternar entre mudanças direcionadas a públicos específicos por longos períodos de tempo. É aqui que entra o “Feature Flags”.

7.5. Chaos engineering como validação de robustez

De forma geral, o Chaos Engineering é uma prática eficiente e poderosa que vem mudando a forma que um determinado sistema é desenvolvido, projetado e executado. Em síntese, é **muito importante na área de TI das empresas**.

Portanto, confira algumas das vantagens da aplicação do Chaos Engineering.
Não é possível saber exatamente como um sistema irá se comportar especificamente em situações do mundo real. A engenharia do caos ajuda a simular os cenários do mundo real de forma específica, abordando o comportamento do sistema e percebendo suas **reações em determinados casos**.

Dessa maneira, é possível se preparar para evitar essas falhas ou amenizar os impactos caso elas ocorram.

Por mais que tentamos nos atentar ao máximo, os sistemas ainda sim estão sujeitos a falhas. Por isso, é essencial compreender **como se comporta o funcionamento de um sistema** e a reação referente a determinados estímulos.

Sendo assim, é possível solucionar possíveis falhas antes que se torne um problema maior. Dito isso, veja os passos de como aplicar o Chaos Engineering na sua empresa.

8. Visualização e Tomada de Decisão

8.1. Dashboards de qualidade

Dashboard é um painel visual que contém informações, métricas e indicadores da empresa. A ideia é que nele estejam representados os números relevantes para a estratégia de negócio e para o alcance dos objetivos organizacionais.

Trata-se de uma ferramenta gráfica que possibilita o monitoramento visual de números, métricas e KPIs. Geralmente, é monitorado por toda a equipe e serve para embasar decisões e acompanhar o desempenho da empresa.

8.2. Ferramentas de BI aplicadas à engenharia

O Business Intelligence tem um grande potencial na engenharia e é capaz de auxiliar dados complexos para tomar decisões embasadas. Além disso, ajuda os engenheiros a identificar padrões, tendências e áreas de melhoria nos processos de produção e gestão.

O que é Business Intelligence?

Business Intelligence refere-se a tecnologias, processos e práticas usados para coletar, integrar, analisar e apresentar informações de negócios. O objetivo do BI é apoiar a tomada de decisões empresariais. Isso é feito através da transformação de dados brutos em informações úteis e insights acionáveis. Ferramentas de BI incluem painéis de controle (dashboards), relatórios, análise de dados e visualização.

Como o BI é utilizado na engenharia?

O Business Intelligence representa um diferencial competitivo significativo na engenharia. Ao transformar dados em insights acionáveis, os engenheiros podem melhorar a eficiência operacional, reduzir custos, mitigar riscos e, acima de tudo, inovar. O Business Intelligence (BI) tem várias aplicações na engenharia, incluindo a gestão de projetos, otimização de processos, gestão de recursos e manutenção da qualidade.

Ele permite o monitoramento em tempo real do progresso dos projetos e a análise de riscos, além de identificar ineficiências e automatizar relatórios, liberando os engenheiros para atividades estratégicas.

Na gestão de recursos, o BI otimiza a alocação de materiais e mão-de-obra, controlando custos. Na manutenção, o monitoramento da qualidade em tempo real e a manutenção preditiva ajudam a manter padrões elevados e reduzir o tempo de inatividade dos equipamentos. Assim, o BI se torna um aliado essencial para a eficiência, segurança e economia na engenharia.

Investir em ferramentas de BI e na cultura de análise de dados é, sem dúvida, um passo estratégico para qualquer empresa que deseja se manter relevante e competitiva no mercado atual. O futuro pertence àqueles que sabem como usar as informações a seu favor!

8.3. Heatmaps de cobertura e falhas

Ele é utilizado em demonstrações gráficas em diversas áreas, como pesquisas relacionadas a dados geográficos, demográficos ou comportamentais.

No mundo digital, ele também é muito importante.

Isso porque é o responsável por identificar padrões de usabilidade e navegação dos usuários.

Diante das informações geradas por um heatmap, você pode criar estratégias baseadas no comportamento dos visitantes.

Em outras palavras, uma grande vantagem estratégica que gera muitas possibilidades para melhoria de desempenho em páginas da web.

Ou seja, falo de uma ferramenta de marketing essencial para empresas que visam à otimização de conversões.

8.4. Indicadores de performance (KPI) e SLA de testes

A definição de SLA. O acrônimo se desenvolve como Service Level Agreement, que, traduzido, significa Acordo de Nível de Serviço.

É um documento que detalha os parâmetros contratuais para execução de uma atividade de prestação de serviço, seja entre uma empresa e seus colaboradores ou dela com algum parceiro comercial para atividades terceirizadas.

Dessa forma, podemos entender que o SLA é o modelo ideal de como o serviço prestado deve ser executado, estabelecendo os níveis de qualidade, eficiência, desempenho, formato, prazos e processos que deverão ser seguidos durante uma atividade de prestação de serviços.

De modo geral, podemos destacar o Service Level Agreement como os parâmetros calculados para uma companhia atingir seus objetivos. Por exemplo, supondo que uma empresa planeje aumentar o faturamento em 50% durante 6 meses, o SLA indica que a cada mês o crescimento deve ser de pelo menos 8,33%.

Obviamente, esse exemplo é bem simplificado, já que o SLA de verdade deve apresentar também como essa meta deve ser batida, assim como os parâmetros e processos utilizados para tal.

Também é importante destacar que o SLA pode ser aplicado em uma grande variedade de setores, atividades e categorias de parceria. Desde que a área tenha uma abordagem única em relação às demais, ela deve contar com um acordo específico para direcionar seus trabalhos.

Nesse caso, a empresa precisa se preocupar em estabelecer os SLA de forma que possam se complementar. Por exemplo, o SLA de atendimento precisa de um escopo único no geral, mas idealmente não conta com parâmetros que entram em conflito com o acordo para o marketing ou a área de vendas.

Um dos parâmetros definidos no Acordo de Nível de Serviço são as métricas mais relevantes para certificar que a performance necessária para alcançar os objetivos está sendo entregue pelos prestadores do serviço.

O Acordo de Nível de Serviço (SLA) aponta quais são as métricas mais importantes (KPI) para demonstrar o real desempenho de uma atividade e se essa performance é adequada para atingir os objetivos estabelecidos.

Essa é a relação básica entre KPI e SLA, considerando que este primeiro acrônimo se refere aos Key Performance Indicators, ou indicadores-chave de desempenho, que servem como um modo de avaliar o desempenho apresentado em determinada área.

Cada área terá indicadores específicos atrelados a ela e idealmente eles devem estar descritos no SLA, garantindo transparência total para a gestão do projeto, além de promover uma monitoria colaborativa.

Aqui vemos também o que é a gestão de KPIs, ou seja, uma demanda específica para definir e acompanhar as métricas mais relevantes para cada área de uma empresa, seja ela atendimento, vendas, logística ou marketing.

9. Conformidade, Ética e Segurança

9.1. Qualidade relacionada à conformidade legal

No desenvolvimento de software deve-se garantir que o produto adere às leis e regulamentos sobre o uso e proteção de dados. Aplica-se a Lei Geral de Proteção de Dados (13.709/2018) e Regulamento Geral sobre a Proteção de Dados (Regulamento da UE 2016/679). É necessário que os artigos da LGPD, e pontos-chave e regras aplicáveis do GDPR sejam lidos e cumpridos, caso contrário, a empresa ou indivíduos por trás do software podem ser sujeitos a processos e multas.

9.2. Boas práticas de segurança como parte da qualidade

Quando se está produzindo um produto, uma grande preocupação é a possibilidade de ataques ou vazamento de dados, para tal planeja-se e segue-se uma abordagem para implementar métodos de segurança. DevSecOps, significando desenvolvimento, segurança e operações, é uma prática que inclui a implementação desde o início do desenvolvimento e integra testes de segurança a todas as etapas do processo de desenvolvimento de software. "software, mais seguro, mais cedo" é o lema da prática. Procurando por possíveis brechas de segurança de acordo com que o código é feito, em contraste a outros métodos que tem segurança como preocupação e etapa final.

9.3. Ética em testes

Ao ter-se dados pessoais, há responsabilidades éticas que devem ser exercidas. Informação sensível não deve ser exposta nem alterada, ao invés, tem-se as boas práticas:

- Dados sensíveis devem ser mascarados ou substituídos por informação fictícia.
- Testes usando dados reais devem ser seguros de forma a evitar qualquer exposição em ambientes isolados.
- Implementar logs e monitoramento para rastrear quem acessou os dados e quando.
- Estabelecer políticas claras sobre a manipulação de dados .
- Treinar testadores e desenvolvedores para se conformar às boas práticas e políticas de proteção de dados.

- Implementar ferramentas de garantia de conformidade

9.4. Testes de acessibilidade como dimensão de qualidade

É ideal que qualquer um possa fazer uso de um serviço, aplicação ou website, o que significa que desenvolvedores devem buscar tornar o produto acessível para casos de deficiência do usuário. Falha em ser acessível afasta usuários do software, logo deve-se testar e se certificar dessa qualidade. Existem muitas ferramentas de teste para este propósito, além de alguns requisitos básicos, em que o software deve:

- Poder ser usado sem o mouse.
- Poder ser usado sem o teclado.
- Poder ser usado sem movimentos precisos.
- Poder ser usado sem limitações no tempo para responder.
- Poder ser usado sem precisar que o usuário efetue ações simultâneas
- Poder ser usado com dispositivos de entrada personalizados

10. Qualidade do Produto Final

10.1. Qualidade percebida na entrega: UX, UI, performance, estabilidade

Quando o produto for entregue ao cliente, a aparência, experiência interagindo com o sistema e desempenho serão as primeiras coisas notáveis. Considerando isso, pode-se criar um certo nível de prioridade na ordem de o que é desenvolvido. O desenvolvedor então pergunta a si mesmo aquilo que o cliente se perguntará quando receber o produto:

- O visual transmite qualidade, modernidade e confiança?
- O sistema faz o que foi prometido?
- Todas as funcionalidades entregues estão completas e corretas?
- O sistema é fácil de entender e usar?
- O sistema responde rapidamente?
- É estável sob carga?
- O sistema responde rapidamente?

10.2. Qualidade em manutenção e extensibilidade do código

Para garantir a longevidade de um sistema, o sistema tem de haver a capacidade para ser mantido e ampliado. Um software de difícil manutenção ou sem escalonamento será abandonado assim que não for mais capaz de sustentar a própria base crescente de usuários ou quando os

usuário ficarem frustrados com um produto que falha com facilidade e requer grandes períodos de tempo para ser consertado.

Dentre alguns meios de garantir a manutenção e extensibilidade do software, há estes básicos:

- Manutenção:
    - Código legível.
    - Documentação.
    - Modularização.
    - Testes automatizados.
- Extensibilidade
    - Adoção de princípios SOLID
    - Abstração
    - Baixo acoplamento e alta coesão

10.3 Suporte técnico e qualidade de atendimento

Usuários podem e irão ter dificuldade com um sistema de software, para isso, tem-se um meio de os usuários se comunicarem com a equipe ou indivíduo por trás do software. Ter um bom atendimento ao cliente para responder perguntas e ajudar com complicações é essencial para uma boa experiência do usuário. Um bom atendimento deve seguir alguns passos de garantia de qualidade:

- Atendimento deve ter disponibilidade e acessibilidade, ou seja, ter bons períodos em que um usuário pode tentar acessar o suporte técnico
- Suporte técnico ágil para responder perguntas
- Comunicação clara entre o técnico e o cliente
- Eficiência em entregar soluções
- Empatia do técnico para com o usuário e suas dificuldades

13.4. Feedback loops com o usuário

Um feedback loop é um ciclo em que é construído a solução para um problema, analisado o quão efetiva e satisfatória essa construção foi e então aprender com a análise e respostas recebidas e teorizar como continuar melhorando, então voltando a construção de melhorias e soluções. Essa prática permite um desenvolvimento e evolução contínua do sistema ao continuar se adaptando às respostas às mudanças e é uma ferramenta indispensável do desenvolvedor, assim sendo em si, uma métrica de qualidade.

10.5. Testes A/B e testes de aceitação com usuários reais

Em testes A/B, duas ou mais variações de design de uma página ou interface, geralmente apenas diferentes em um único quesito ou componente, são apresentados a visitantes, para estudar a performance de cada opção de forma a escolher a que traz a melhor experiência de usuário. Este

método de teste quando implementado permite maximizar a qualidade de UX, e logo, sendo uma ferramenta recomendada.

11. Referências

MENDES BUENO Leonardo. Gurus da Qualidade: David A. Garvin. **Qualiex Forlogic**, 19, set. 2023. Disponível em: https://blogdaqualidade.com.br/gurus-da-qualidade-david-a-garvin/. Acesso em: 19 jun. 2025.

SARTORI Adriana. Os 14 princípios de Deming e seu impacto no conceito de qualidade. **Qualyteam,** 14, mai. 2024. Disponível em: https://qualyteam.com/pb/blog/14-principios-deming/. Acesso em: 19 jun. 2025.

LEITE Gabriel. Gurus da Qualidade: Joseph Moses Juran. **Qualiex Forlogic**, 01, ago. 2019. Disponível em: https://blogdaqualidade.com.br/gurus-da-qualidade-joseph-moses-juran/. Acesso em: 19 jun. 2025.

O QUE É: Custo da Não Qualidade. **Cirius Quality,** 22, set. 2023. Disponível em: https://ciriusquality.com.br/glossario/o-que-e-custo-da-nao-qualidade/?srsltid=AfmBOoo-liGn0LXNyo1gnOOt-Q77QyGkn8TrLxtHPKkzhR0xGNmaw67w. Acesso em: 18 jun. 2025.

AMARAL CASTRO Bruna. Qualidade do produto e qualidade do processo, qual a diferença. **Zeev by stoque**, 31, mar. 2025. Disponível em: https://zeev.it/blog/qualidade-produto-qualidade-processo/. Acesso em: 16 Jun. 2025.

BROGNOLI Arthur. Qualidade percebida: o que o cliente enxerga na sua solução. **Clint Digital,** [s.d]. Disponível em: https://www.clint.digital/artigos/qualidade-percebida-o-que-o-cliente-enxerga-na-sua-solucao. Acesso em: 15, jun. 2025.

AMCHAM BRASIL. Metodologias ágeis: como transformar a cultura da sua empresa. *Amcham Brasil*, 01, ago. 2023. Disponível em: https://www.amcham.com.br/blog/metodologias-ageis. Acesso em: 17 jun. 2025.

WHAT are the advantages and disadvantages of ISO 25010 compared to other software quality models? LinkedIn, [s.d.]. Disponível em: https://www.linkedin.com/advice/1/what-advantages-disadvantages-iso-25010-compared?lang=pt&lang=pt&originalSubdomain=pt. Acesso em: 17 jun. 2025.

CAVALCANTI, Dani. NBR ISO 12207: processos de ciclo de vida do software. *LinkedIn*, [s.d.]. Disponível em: https://www.linkedin.com/pulse/nbr-iso-12207-processos-de-ciclo-vida-do-software-dani-cavalcanti/. Acesso em: 17 jun. 2025.

EVOKE TECHNOLOGIES. New software testing standards. *Evoke Technologies Blog*, [s.d.]. Disponível em: https://www.evoketechnologies.com/blog/new-software-testing-standards/. Acesso em: 18 jun. 2025.

PROMOVE SOLUÇÕES. Níveis de maturidade do CMMI. *Promove Soluções*, [s.d.]. Disponível em: https://promovesolucoes.com/niveis-maturidade-cmmi/. Acesso em: 18 jun. 2025.

SOFTEX. MPS.BR – Melhoria de Processo do Software Brasileiro. *Softex*, [s.d.]. Disponível em: https://softex.br/mpsbr/. Acesso em: 19 jun. 2025.

DQS DO BRASIL. Certificação ISO/IEC 15504 – SPICE. *DQS Global*, [s.d.]. Disponível em: https://www.dqsglobal.com/pt-br/certifique/certificacao-iso-iec-15504-spice. Acesso em: 19 jun. 2025.

TIFLUX. ITIL: dicas de ouro para serviços de TI. *Tiflux Blog*, [s.d.]. Disponível em: https://tiflux.com/blog/itil-dicas-de-para-servicos-de-ti/. Acesso em: 19 jun. 2025.

VISURE SOLUTIONS. Requirements verification and validation. *Visure Solutions*, [s.d.]. Disponível em: https://visuresolutions.com/pt/alm-guide/requirements-verification-and-validation/. Acesso em: 19 jun. 2025.

LINKEDIN. How can you effectively manage ambiguity in operations? *LinkedIn*, [s.d.]. Disponível em: https://www.linkedin.com/advice/3/how-can-you-effectively-manage-ambiguity-operations?lang=pt&originalSubdomain=pt. Acesso em: 19 jun. 2025.

PRODUTTIVO. Gestão da qualidade em serviços: o que é, importância e como aplicar. *Produttivo Blog*, [s.d.]. Disponível em: https://www.produttivo.com.br/blog/gestao-da-qualidade-em-servicos/. Acesso em: 20 jun. 2025.

VISURE SOLUTIONS. Requirements traceability matrix. *Visure Solutions*, [s.d.]. Disponível em: https://visuresolutions.com/pt/alm-guide/requirements-traceability-matrix/. Acesso em: 20 jun. 2025.

REVISTA TSPI. Testes baseados em riscos (Risk-Based Testing). *Medium*, [s.d.]. Disponível em: https://medium.com/revista-tspi/testes-baseados-em-riscos-risk-based-testing-b7dfa751ec17. Acesso em: 20 jun. 2025.

VISURE SOLUTIONS. Model-based testing. *Visure Solutions*, [s.d.]. Disponível em: https://visuresolutions.com/pt/alm-guide/model-based-testing/. Acesso em: 20 jun. 2025.

DEVMEDIA. Desenvolvimento orientado por comportamento (BDD). *DevMedia*, [s.d.]. Disponível em: https://www.devmedia.com.br/desenvolvimento-orientado-por-comportamento-bdd/21127. Acesso em: 19 jun. 2025.

DEVMEDIA. Testes exploratórios: teoria e prática. *DevMedia*, [s.d.]. Disponível em: https://www.devmedia.com.br/testes-exploratorios-teoria-e-pratica/22791. Acesso em: 19 jun. 2025.

EMBARCADOS. O sucesso da IoT depende de testes de RF. *Embarcados*, [s.d.]. Disponível em: https://embarcados.com.br/o-sucesso-da-iot-depende-de-testes-de-rf/. Acesso em: 19 jun. 2025.

QUALYTEAM. Garantia da qualidade: o que é, importância e como aplicar. *Qualyteam Blog*, [s.d.]. Disponível em: https://qualyteam.com/pb/blog/garantia-da-qualidade/. Acesso em: 19 jun. 2025.

LUCIDCHART. O que é matriz RACI e como aplicá-la em seus projetos. *Lucidchart Blog*, [s.d.]. Disponível em: https://www.lucidchart.com/blog/pt/matriz-raci#:~:text=O%20que%20%C3%A9%20matriz%20RACI,projeto%20s%C3%A3o%20mapeados%20por%20completo. Acesso em: 19 jun. 2025.

FONSECA, Bruna. User Story, critério de aceite e regras de negócio. *Bruna Fonseca*, [s.d.]. Disponível em: https://www.brunafonseca.pro.br/post/user-story-crit%C3%A9rio-de-aceite-e-regras-de-neg%C3%B3cio. Acesso em: 19 jun. 2025.

DEVMEDIA. Gerência de configuração e mudanças. *DevMedia*, [s.d.]. Disponível em: https://www.devmedia.com.br/gerencia-de-configuracao-e-mudancas/31327. Acesso em: 19 jun. 2025.

JETBRAINS. Build and release management – FAQ. *JetBrains TeamCity*, [s.d.]. Disponível em: https://www.jetbrains.com/pt-br/teamcity/ci-cd-guide/faq/build-release-management/#. Acesso em: 19 jun. 2025.

PARIPASSU. Cultura da qualidade: o que é, importância e como aplicar. *Paripassu Blog*, [s.d.]. Disponível em: https://www.paripassu.com.br/blog/cultura-da-qualidade. Acesso em: 19 jun. 2025.

MONITORATEC. Comunicação QA: como melhorar a comunicação entre times de qualidade e desenvolvimento. *Monitoratec Blog*, [s.d.]. Disponível em: https://www.monitoratec.com.br/blog/comunicaao-qa/. Acesso em: 19 jun. 2025

ALURA. Agile Testing: o que é e qual o papel do QA num time ágil? *Alura*, [s.d.]. Disponível em: https://www.alura.com.br/artigos/agile-testing-o-que-e-qual-papel-qa-num-time-agil. Acesso em: 19 jun. 2025.

TESTING COMPANY. Gestão do conhecimento e suas relações com documentação de software. *Testing Company Blog*, [s.d.]. Disponível em: https://testingcompany.com.br/blog/gestao-do-conhecimento-e-suas-relacoes-com-documentacao-de-software. Acesso em: 19 jun. 2025.

OBJECTIVE. Teste contínuo: o que é, benefícios e como aplicar. *Objective Insights*, [s.d.]. Disponível em: https://www.objective.com.br/insights/teste-continuo/. Acesso em: 19 jun. 2025.

SUKLA, Rakesh. What is continuous quality? *Medium*, [s.d.]. Disponível em: https://medium.com/@rakesh.sukla53/what-is-continuous-quality-617ca0196e61. Acesso em: 19 jun. 2025.

RED HAT. Shift Left vs. Shift Right. *Red Hat*, [s.d.]. Disponível em: https://www.redhat.com/pt-br/topics/devops/shift-left-vs-shift-right. Acesso em: 19 jun. 2025.

OBJECTIVE. Pipeline DevOps: o que é, etapas e como implementar. *Objective Insights*, [s.d.]. Disponível em: https://www.objective.com.br/insights/pipeline-devops/. Acesso em: 19 jun. 2025.

PEREIRA, Diego. Entrega progressiva: Canary e Feature Flags. *LinkedIn*, [s.d.]. Disponível em: https://www.linkedin.com/pulse/entrega-progressiva-canary-e-feature-flags-diego-pereira/. Acesso em: 19 jun. 2025.

MYTECH. Chaos Engineering: o que é, como funciona e como aplicar. *MyTech Blog*, [s.d.]. Disponível em: https://www.mytech.com.br/blog/chaos-engineering/. Acesso em: 19 jun. 2025**.**

PATEL, Neil. Dashboard: o que é, para que serve e como criar um. *Neil Patel Blog*, [s.d.]. Disponível em: https://neilpatel.com/br/blog/dashboard-o-que-e/. Acesso em: 19 jun. 2025.

PONTA ENGENHARIA. Business Intelligence: como um aliado na engenharia. Ponta Engenharia Blog, [s.d.]. Disponível em: https://pontaengenharia.com.br/blog/business-intelligence-como-um-aliado-na-engenharia. Acesso em: 19 jun. 2025.

PATEL, Neil. Heatmap: o que é, para que serve e como usar. *Neil Patel Blog*, [s.d.]. Disponível em: https://neilpatel.com/br/blog/heatmap-o-que-e/. Acesso em: 19 jun. 2025**.**

ZENDESK. KPI e SLA: o que são e qual a diferença entre eles? *Zendesk Blog*, [s.d.]. Disponível em: https://www.zendesk.com.br/blog/kpi-sla/. Acesso em: 19 jun. 2025.

PM3. Feedback loop: o que é, etapas e importância para Produto. *SOMOS 3 INTERNET S.A,* 20 mai. 2022. Disponível em: https://pm3.com.br/blog/feedback-loop-o-que-e-etapas-importancia/ . Acesso em: 12 jun. 2025.

Tim Neusser. A/B Testing 101. *Nielsen Norman Group,* 30 ago. 2024. Disponível em: https://www.nngroup.com/articles/ab-testing/. Acesso em: 12 jun. 2025.

Programae. Qual é a importância do suporte técnico para software?. *Desenvolvimento de Software, 1*8 out. 2024. Disponível em: https://programae.org.br/software/qual-e-a-importancia-do-suporte-tecnico-para software/?srsltid=AfmBOoowgTYDU3UWAg0icYmHuvCkJieEyThT4u1eYH1licx7tqvoi1sw. Acesso em: 12 jun. 2025.

Lyncas. Manutenção de software: garanta o funcionamento e a eficiência contínua. Lyncas Tecnologia da Informação Ltda., 20 jun. 2024. Disponível em: https://lyncas.net/blog/manutencao-de-software-eficiencia-continua/#:~:text=A%20manuten%C3%A7%C3%A3o%20de%20software%20%C3%A9,e%20preven%C3%A7%C3%A3o%20de%20problemas%20futuros. Acesso em: 12 jun. 2025.

tropical hub. TECNOLOGIA DA INFORMAÇÃO: O QUE É A EXTENSIBILIDADE DE UM SOFTWARE?. *Tropical Hub, [*s.d.]. Disponível em: https://www.tropicalhub.co/blog/tecnologia-da-informacao-extensibilidade-de-software. Acesso em: 19 jun. 2025.

Fausto Reichert. TTFV, Time to First Value: o que é o primeiro valor ao cliente?. *piperun,* [s.d.]. Disponível em: https://crmpiperun.com/blog/time-to-first-value-

ttfv/#:~:text=Clientes%20satisfeitos%20logo%20no%20in%C3%ADcio,clientes%20quanto%20a%20pr%C3%B3pria%20organiza%C3%A7%C3%A3o. Acesso em: 12 jun. 2025.

Virtual Vision. Acessibilidade Digital: Garantindo a Inclusão de Todos. *virtualvision,* 25 jul. 2023. Disponível em: https://www.virtualvision.com.br/blog/acessibilidade-digital/. Acesso em: 12 jun. 2025.

geeksforgeeks. The Ethics of Software Testing: Avoiding Biases and Ensuring Fairness. *Sanchhaya Education Private Limited,* 11 dez. 2023. Disponível em: **https://www.geeksforgeeks.org/software-testing/ethics-of-software-testing-avoiding-biases-and-ensuring-fairness/.** Acesso em: 12 jun. 2025.

Red Hat. O que é DevSecOps?. *Red Hat, inc,* 22 fev. 2023. Disponível em: https://www.redhat.com/pt-br/topics/devops/what-is-devsecops. Acesso em: 12 jun. 2025.

Juliana de Oliveira Soares. O que é gerenciamento de conformidade legal e por que ele é uma necessidade nas empresas?. *Rocha Cerqueira, 13 mar. 2025.* Disponível em: https://rochacerqueira.com.br/o-que-e-gerenciamento-de-conformidade-legal-e-por-que-ele-e-uma-necessidade-nas-empresas/. Acesso em: 12 jun. 2025.

Presidência da República. L13709. *planalto.gov.* 14 ago. 2018. Disponível em: https://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/l13709.htm. Acesso em: 12 jun. 2025.

Intersoft Consulting. GDPR. *Intersoft Consulting,* [s.d.]. Disponível em: https://gdpr-info.eu/. Acesso em: 12 jun. 2025.

Conclusão

A qualidade de software não é apenas um diferencial competitivo — é um compromisso com a eficiência, segurança e satisfação do usuário. Este manual técnico suplementar buscou reunir os principais fundamentos, práticas e ferramentas que sustentam esse compromisso, servindo como guia para equipes que almejam excelência em seus projetos. Ao colocar a qualidade no centro do desenvolvimento, pavimenta-se o caminho para soluções mais robustas, sustentáveis e alinhadas às reais necessidades do mercado.