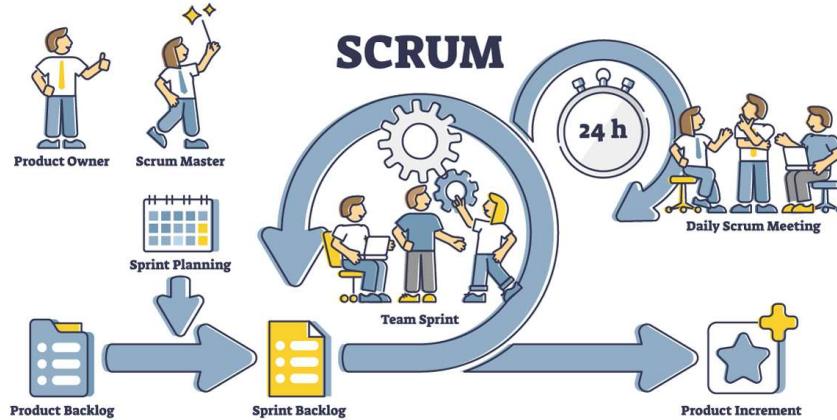


## NÃO PODE FALTAR

# GESTÃO DA QUALIDADE

Daniela Teresa Rossignoli Uebele

Ver anotações



Fonte: Shutterstock.

### Deseja ouvir este material?

Áudio disponível no material digital.

#### PRATICAR PARA APRENDER

Caro aluno, quando buscamos por qualidade, logo pensamos em custo alto. Creio que seja uma relação parcialmente equivocada, no entanto, qualidade no desenvolvimento de software é, hoje, um requisito mercadológico indiscutivelmente essencial.

No ponto de vista do seu cliente, bem pago, para o desenvolvimento do sistema, já inclui, impreterivelmente, a questão da qualidade. Quando entende que a necessidade dele é um sistema que gerenciará ou otimizará os processos de sua empresa, é porque ela se encontra em um momento de estagnação e necessita de um diferencial, ou seja, de QUALIDADE, que tem que fazer parte do pacote do software a ser desenvolvido, bem como muitas análises de riscos, a fim de que ajude o controle de cada um deles, além do comprometimento com o tempo, o custo e o escopo do produto.

A empresa contratante do desenvolvimento de um sistema personalizado espera que seu novo projeto lhe traga diferenciais estratégicos ou resolva problemas recorrentes. Não estamos falando de mais um sistema e sim daquele que manterá a empresa competitiva no mercado em que atua, talvez até revolucionar a área de atuação.

Portanto, visando a atender às expectativas e aos anseios de uma organização, a gestão da qualidade deve ser implantada no processo de planejamento, execução e controle dessa qualidade esperada. Sendo assim, faz-se necessária a implantação de metodologias de projetos buscando meios de se definir, medir e controlar o sistema de qualidade do Software.

A equipe de desenvolvimento e o cliente, por meio dos documentos gerados no processo de iniciação do projeto, do termo de abertura, do planejamento das atividades, dos riscos, da comunicação e de todas as 10 áreas do conhecimento trabalhadas pelo PMBOK, terão de definir como garantir a qualidade. Para tanto, existem inúmeras normativas que podem ser acrescidas nesse processo, aqui veremos a ISO 25010.

Serão trabalhados modelos de qualidade para se buscar ferramentas já utilizadas, facilitando esse processo de definição de pronto, sendo visto como um fator de qualidade. Será por meio deles e de outras técnicas que será possível traçar métricas, definir o que pode ser dito como qualidade no desenvolvimento de um software, já que este não sofre desgaste, porém necessita de muito planejamento para uma longa vida útil; sua manutenção deverá ser eficiente e eficaz; a segurança do sistema, os dados e as informações deverão ser garantidos, como muitas outras variáveis que serão estudadas.

Voltemos a estudar o projeto em que você e a sua equipe terão que desenvolver um sistema. A fim de atender à área da saúde, essa ferramenta será utilizada dentro da sala de cirurgia para que os médicos possam manipular radiografias de raio-X, tomografias e ultrassonografias sem a necessidade de remover as luvas, portanto, serão utilizados recursos tecnológicos de identificação de movimentos e/ou voz com a junção de realidade virtual, visando à minimização do tempo do procedimento cirúrgico, pois o profissional será poupadão de ter que, novamente, higienizar as mãos para voltar ao procedimento, trazendo benefícios também ao paciente que passará menos tempo anestesiado, liberando o centro cirúrgico para outros procedimentos.

Para promover a qualidade necessária, visando à utilização de sistema de qualidade de software, faz-se necessária a criação dos requisitos funcionais necessários para o desenvolvimento do sistema, criando, a partir desses requisitos, a definição dos requisitos não funcionais, que facilitará a análise dos aspectos externos e internos do desenvolvimento do sistema. Com essas informações definidas, é viável a criação dos critérios de qualidade para cada um dos requisitos funcionais e não funcionais, visando à geração da definição de pronto (DoD).

Utilizando as características definidas pela ISO 25010 e suas subcaracterísticas, verifique as aplicáveis aos sistemas e como seriam efetuados os testes para responder os questionamentos, a fim de se verificar se o sistema desenvolvido está atendendo a tais características sugeridas.

Esses processos devem envolver a equipe de desenvolvimento, portanto, solicitar a ajuda de todos os envolvidos será de grande importância para o sucesso da busca pela qualidade. Crie o quadro com o conceito de Kanban, com 3 colunas divididas em TO DO (Fazendo), *Work in Progress* (WIP) desenvolvendo e DONE (pronto); insira as atividades a serem executadas inicialmente e escreva, logo abaixo, os critérios de pronto, assim, tornaremos transparente o processo de atividade concluída.

A busca por qualidade e a compreensão dos conceitos e das formas de aplicá-la no desenvolvimento do software levarão você a ser um profissional não apenas capacitado, com habilidades técnicas, mas com qualidade.

## CONCEITO-CHAVE

### ■ GESTÃO DA QUALIDADE

**QUALIDADE!** O que essa palavra representa para você? Qualidade pode ser observada por inúmeros olhares. Buscando uma interpretação ampla nos dicionários, é definida pela forma como se pode distinguir pessoas e objetos devido às suas propriedades e aos seus atributos de forma a possuir diferenciação significativa segundo a sua natureza, ou seja, tudo que fazemos ou criamos pode e deve ter qualidade. Se recorrermos ao conteúdo da seção anterior, a mitigação de riscos compreende uma das formas de se buscar a qualidade, já que, no momento em que se reduzem os riscos que levam ao atraso ou aumento dos custos, talvez até à mudança do escopo, já temos a perda da qualidade, e essa compreensão nos permite uma análise ampla sobre a qualidade de um projeto de desenvolvimento de software (COSTA; PEREIRA, 2019).

A fim de gerirmos a qualidade, observaremos, no estudo desta seção, que a qualidade tem inúmeras formas de ser mensurada e que existe a possibilidade de se **adequar essas métricas em diferentes metodologias de projeto**.

Ao analisar o contexto de qualidade do ponto de vista do mercado de desenvolvimento de software, que é altamente competitivo, e levando em conta que na década de 1950 não se imaginava o quanto importante seria um software, possuindo a imensa capacidade de manipular um enorme volume de dados e gerando informações de alto grau de complexidade, comprovando a Lei das "consequências não pretendidas" (PRESMAN, 2006).

Portanto, o gerenciamento da qualidade inclui analisar os aspectos tangíveis (aspectos que são tocados / visíveis / usado / mensurado) e intangíveis (os valores que não podem ser tocados ou medidos), e depende diretamente da relação de entrega do sistema e/ou dos serviços que é capaz de oferecer, ou seja, um produto diferenciado é fundamental para fortalecer o nome da empresa desenvolvedora.

Dentro do atual sistema mercadológico não se fala apenas em entrega de produtos dentro do prazo e custo, que funciona conforme as expectativas do cliente, mas também em vender uma experiência, de forma que ela se torne inesquecível, ofertando ao seu cliente qualidade do início ao fim do processo de venda, o que nos leva a mais um questionamento: como garantir a qualidade em projetos de desenvolvimento de software? Podemos pensar da mesma forma que pensamos ao buscar a qualidade para qualquer outro projeto? Resposta: utilizando-se a gestão do triângulo de ferro, conservando o custo, o tempo e o escopo do projeto, conforme previsto, assim, atendendo à qualidade no que diz respeito ao aspecto tangível, aquele que foi previamente acordado mediante as restrições iniciais de qualquer projeto (COSTA; PEREIRA, 2019).

### ■ IMPORTÂNCIA DA QUALIDADE

Para entender a importância que a qualidade vem tomando desde os anos 1980.

Existem inúmeras instituições reconhecidas que disponibilizam normas para permitir o planejamento, desenvolvimento e controle da qualidade, inclusive específicas para softwares. Vamos citar algumas normas nacionais e internacionais:

- NBR 13596: versão nacional da ISO 9126 falando sobre a qualidade de produtos de software, fazendo parte da família ISO 9000.
- NBR ISO 9001: padronização para a garantia da qualidade em projetos, desenvolvimento, instalação e processos – Sistema de qualidade.
- NBR ISO 9000-3: utilização da ISO 9000 no processo de desenvolvimento de software, ou seja, gestão e garantia da qualidade.
- ISO 15498: é um guia que traz diretrizes para avaliação dos produtos de software, criado a partir da ISO 9126.
- ISO 12119: desenvolvido para os softwares genéricos, vendidos em prateleiras, em que a empresa se adequa ao produto e não o produto à empresa; normas que determinam as características de qualidade de pacotes de software.
- IEEE P1061: *Standard for Software Quality Metrics Methodology* - normativas para tratar métodos de padronizar a qualidade de software, incluindo algumas formas de medição.
- CMMI *Capability Maturity Model Integration*: Modelo da SEI (Instituto de Engenharia de Software do Departamento de Defesa dos USA); não se enquadra nas normas ISS, porém é amplamente aceita para a avaliação da qualidade dos processos de desenvolvimento do software;
- SPICE ISO 15504 ou apenas SPICE: melhora os processos de desenvolvimento de software, semelhante a ISO 12207, referente à parte de ciclo de vida de Software.

Além dessas, existem muitas outras, por isso a importância de um amplo conhecimento sobre o que é qualidade, como aplica-la, controlar, medir. São inúmeros contextos entrelaçando planejamento, execução e monitoramento (WEBER *et al.*, 2015).

#### EXEMPLIFICANDO

A busca por qualidade no desenvolvimento de software é bem visível quando falamos sobre os sistemas web e os apps dos bancos; não basta nos oferecer os serviços e recursos necessários para a movimentação financeira. Os clientes, no início, tiveram uma grande resistência a sua utilização, afinal, não tinham como saber se o acesso via rede mundial era seguro. Foi necessário mais que um software bem desenvolvido; a segurança e os meios de demonstrar ao cliente que o banco também se preocupava com esse aspecto intangível foram essenciais, pois só era possível mensurar a inexistência de falhas de segurança – sinônimo de qualidade.

o

Ver anotações

Tudo isso se transformou em um ciclo de “não uso, pois não sei se é seguro” e “preciso que usem para comprovar a qualidade do sistema mediante a segurança”. Para isso, a maioria dos bancos, inicialmente, investiu na divulgação; se houvesse problema, o banco assumiria a responsabilidade. Após esse processo, a busca, hoje, está pelas funcionalidades, pelo fácil acesso, pelo rápido entendimento e pelo acesso eficiente.

Ver anotações

## PROCESSOS RELACIONADOS AO GERENCIAMENTO DA QUALIDADE DO PROJETO

A principal preocupação quanto ao gerenciamento da qualidade durante o desenvolvimento de software é a busca por garantir que o sistema de software a ser desenvolvido seja “adequado para seus objetivos”, atendendo às necessidades dos usuários, proporcionando confiabilidade, usabilidade e eficiência em seu funcionamento. Nos últimos 20 anos, a utilização da gestão da qualidade, de novas tecnologias e métodos de testes por meio de softwares trouxe melhorias importantes para o aumento da qualidade do produto desenvolvido (SOMERVILLE, 2018).

O PMBOK de 2018 indica três processos relacionados ao gerenciamento da qualidade do projeto:

- Planejar o gerenciamento da qualidade.
- Gerenciar a Qualidade.
- Controlar a Qualidade.

## MÉTRICAS DA QUALIDADE

Para alcançar uma compatibilidade com a qualidade, é importante criar meios de medi-la. A satisfação do cliente está no atendimento de suas expectativas, portanto, compreender, controlar e influenciar as necessidades dos clientes são caminhos para atender à qualidade desses *stakeholders*. A gestão da empresa possui a responsabilidade de garantir a qualidade, uma vez que deve fornecer recursos adequados para que o planejamento seja executado. A qualidade do produto não está desvinculada dos interesses organizacionais, como o custo e prazo (COSTA; PEREIRA, 2019).

A melhoria contínua é uma necessidade do ciclo conhecido como PDCA, que significa:

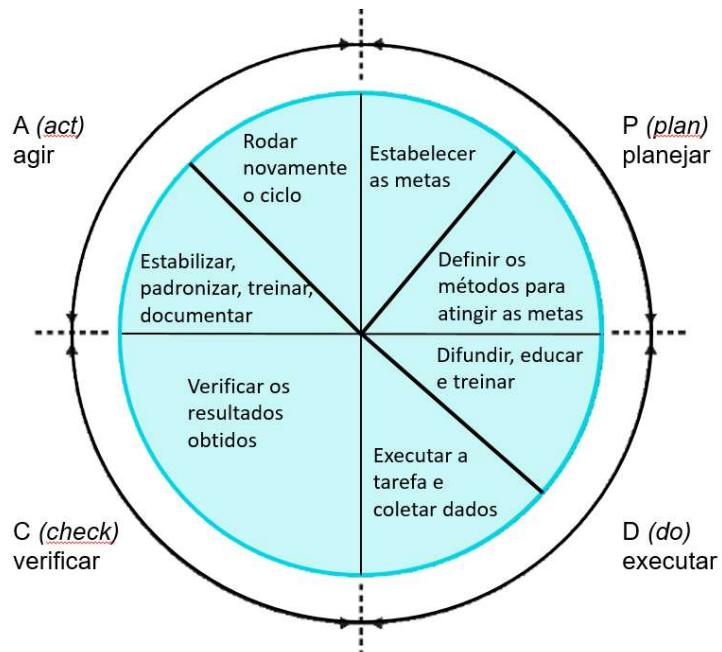
**Plan** = planejar – Estabelece metas e define os métodos necessários para se obter os objetivos do processo.

**Do** = fazer – Executa, difundi, treina, educa as tarefas e coletas os dados.

**Check** = verificar – Faz a verificação dos resultados.

**Act** = agir – Realiza ações corretivas utilizando-se de padronizações, treinamentos e documentos dos processos.

Figura 3.3 | Ciclo PDCA: planejar (*plan*), executar (*do*), verificar (*check*) e agir (*act*)



Ver anotações

Fonte: Carvalho (2018, p. 187).

Esse ciclo deve ser utilizado constantemente e a qualquer momento do projeto, pois o objetivo de melhoria contínua exige essa constância; além disso, esse recurso se utiliza de outras ferramentas capazes de trazer meios de se medir a qualidade (CARVALHO, 2018).

#### GERENCIAMENTO DA QUALIDADE TOTAL (GQT)

A utilização do **gerenciamento da qualidade total** (GQT) visa à conscientização da qualidade em todos os processos de uma empresa, incluindo os *stakeholders* externos (fornecedores, distribuidores e parceiros); o uso do **seis Sigma** ( $6\sigma$ ), que se refere à eliminação de defeitos por um conjunto de boas práticas sistêmicas na melhoria de processos (DMAIC - definir, medir, analisar, melhorar e controlar), visa a resolver problemas de causa desconhecida; e **Lean Seis Sigma**, que é a junção da metodologia seis sigma com as ferramentas e os conceitos do **Lean Manufacturing**, traz a melhoria dos resultados, minimizando as variações nos processos, eliminando desperdícios e as atividades que não agregam valor ao projeto, bem como aprimora o gerenciamento e a qualidade do produto, pois, para o desenvolvedor, as características internas do desenvolvimento do sistema está na legibilidade, testabilidade e eficiência da criação e manutenção dos sistemas desenvolvidos. A prevenção em vez da inspeção é uma necessidade real, pois o custo do retrabalho é muito maior do que a prevenção da existência de falhas (SOMERVILLE, 2018).

Uma das principais ferramentas para o processo de garantia da qualidade é a auditoria, permitindo uma revisão totalmente independente e benefícios como: as lições aprendidas que trazem benefícios para os futuros e atuais projetos; visualização de lacunas, não conformidades e deficiências do

planejamento; avaliação dos fornecedores por meio das verificações de atendimento dos requisitos; e a melhoria contínua da organização por meio do giro do ciclo (PDCA) (COSTA; PEREIRA, 2019).

#### GESTÃO DA QUALIDADE FORMAL

Observe, mais uma vez, que a qualidade não está embasada exclusivamente na entrega do software dentro do prazo, do custo e do escopo predefinido, mas é importante que se tenham boas práticas durante o desenvolvimento, que sejam criadas documentações adequadas, não apenas um manual do usuário, e linhas de código documentadas, bem como se tenha uma organização estrutural das bibliotecas de desenvolvimento e que haja utilização das boas práticas para a análise antes mesmo da codificação, gerando um sistema de fácil manutenção.

Portanto, quando se pensa em equipes de desenvolvimento de um software, logo vem à mente a ideia de um sistema dito como grande, devido à quantidade de funcionalidades ou mesmo por se utilizar de tecnologias inovadoras, já que tais sistemas, possivelmente, terão longa vida de utilização e levarão alguns anos para ser concluídos. Para esses casos, a gestão da qualidade formal é crucial para o sucesso do projeto, e isso demandará a necessidade de recursos compartilhados entre todos os membros, oferecendo um recurso de acesso constante e documentações para se definir o que é qualidade (CARVALHO, 2018).

Existe uma certa dificuldade em definir se um sistema cumpre ou não as especificações. A interpretação dos requisitos pode ser equivocada em sua descrição tanto pelo cliente quanto pelos desenvolvedores, tornando o processo de descrição dos requisitos equivocado e incompleto. Por existir inúmeros stakeholders envolvidos no projeto, é provável que alguns requisitos fiquem fora do desenvolvimento, dessa forma, para alguns, o sistema será de baixa qualidade. O processo de manutenção, por exemplo, não dá para ser totalmente mensurado, o que torna inviável a medição de algumas características do sistema desenvolvido (SOMERVILLE, 2018).

Para uma exemplificação de métricas de qualidade, pode-se criar uma tabela utilizando ferramentas de planilha eletrônica ou softwares específicos, desde que estes permitam o acesso compartilhado com todos os membros. O exemplo para visualização será feito no formato retrato, sendo melhor desenvolvido em paisagem, pois não haverá apenas 2 requisitos como no exemplo do Quadro 3.9.

Quadro 3.9 | Métricas de qualidade de sistemas que se utilizam de cadastros e relatório

Métricas da Qualidade    14/12/2020    Plano de responsabilidade\*

Cód.	1	2
Requisito	Consistência e validação dos dados a serem inseridos ou alterados.	Criação de relatórios de demanda de produtos no e-commerce.

o

Ver anotações

Cód.	1	2
Indicador	IQ01 – verificação das validações de tipo de dado e sua validade no momento da digitação ou finalização do campo.	IQ02 – Tempo médio de criação do relatório.
Tolerância	>= 99%	<=30 segundos
Métodos de medição	Testes manuais. Verificação de código utilizando bibliotecas e API's predefinidos pela equipe. O usuário, ao utilizar o sistema, verifica as entradas de dados.	Cronômetro por meio de sistemas automatizados do tempo de criação do relatório com sobrecarga de dados reais. $\Sigma$ Tempo de criação do relatório / total de relatórios gerados.
Periodicidade	Diária	Diária
Quem mede	Equipe de desenvolvimento, usuário final.	Equipe de desenvolvimento, usuário final.
Doc. de registro	Formulário FIQ01	Formulário FIQ01
Ação corretiva*	Orientar os atendentes para atenção ao processo de cadastro. Anotação de qualquer tipo de falha.	Identificar os relatórios com maior tempo de criação e verificar o processo de melhoria na criação do relatório.
Quando medir*	A cada medição.	A cada medição.
Responsável*	Gestor do projeto e equipe desenvolvedora.	Supervisor de qualidade, equipe desenvolvedora.

Fonte: adaptado de Carvalho (2018).

#### REFLITA

Você já pesquisou os bugs que causaram não apenas perdas financeiras, mas também perdas humanas? No mundo moderno da informação e da informatização, erros computacionais são, por vezes, destruidores. Existem alguns bugs que são realmente assustadores, e as falhas estão ligadas à falta de métodos de análise do sistema, de planejamento de testes e utilização de métricas de qualidade, o que leva à falta de qualidade no processo de desenvolvimento. O erro do hardware foi, por muitas vezes, um dos mais preocupantes, porém já se vão alguns anos que o software tem apresentado preocupações maiores.

O processo de qualidade para todo e qualquer sistema é de fundamental importância, pois os prejuízos financeiros, psicológicos e de saúde também podem ser causados por consequência de sistemas mal projetados, testados e entregues com descasos.

o  
Ver anotações

## EVOLUÇÃO DOS SISTEMAS DE QUALIDADE DE SOFTWARE

No final dos anos 1980, surgiu o método de programação orientada a objetos; já em 1997, surgiu o método de análise *Unified Modeling Language* (UML), amplamente utilizado por permitir que a análise do sistema seja desenhada e discutida entre a equipe de gestão e de desenvolvimento, reduzindo o tempo de trabalho de codificação e do retrabalho. Em 2001, por sua vez, a criação do manifesto ágil trouxe a flexibilização do enrijecimento causado nos anos anteriores, em busca da qualidade na produção de sistemas de software com a visão equivocada embasada no conceito da qualidade de um produto de linha de produção; por fim, a partir de 2002, inúmeros processos ágeis foram sugeridos para que o cliente pudesse interagir no decorrer do desenvolvimento do software, buscando-se o alinhamento e o dinamismo do escopo com o mercado e, por consequência, a entrega de um sistema funcional, visando ao sistema de qualidade de software (ROCHA, 2001; WEBER, 2008; SOMERVILLE, 2018).

Dentro dessa evolução em busca de **sistemas de qualidade de software**, passou-se a existir novas formas de se trabalhar. A metodologia Scrum surgiu para agregar mais produtividade nos processos; o conceito Lean visa a eliminar toda atividade que não é prioridade, evitando o comprometimento do cronograma; já o conceito cascata visa a um controle de processo linear de forma rigorosa, começando uma nova etapa apenas quando a anterior está concluída, opondo-se à metodologia ágil, que visa ao próprio projeto ou produto, permitindo melhorias e mudanças com maior frequência, baseadas no feedback do cliente. E além dos métodos gerenciais, existem inúmeras linguagens para desenvolver sistemas adequados a cada cliente, com muita personalização e personificação, como: sistemas web, *web services*, *cloud computing* (software como serviço); aplicativos mobile, IoT (*Internet of Things*) entre outros.

Os inúmeros processos de desenvolvimento de software utilizam atividades como: **análise de requisitos** – momento em que se toma conhecimento do real problema para o desenvolvimento do sistema; **design do software** – momento de planejar a solução do problema apontado na documentação dos requisitos; **código** – transforma em código o que foi planejado no designer do sistema; e **teste** – a busca por defeitos/falhas e um meio de se medir e controlar a qualidade do sistema desenvolvido (JALOTE, 2005).

Nesse processo de análise de requisitos, a metodologia UML é uma excelente forma de se modelar (desenhar) e documentar analisando os requisitos funcionais, ou seja, quais são as funcionalidades que o seu sistema terá de oferecer ao usuário. Exemplo de requisitos funcionais para um consultório de atendimento médico:

- RF01 – O sistema deve gerar a lista de consultas do dia para cada um dos médicos.
- RF02 – Os usuários devem ser identificados por um código de 6 dígitos.
- RF03 – As secretárias podem fazer pesquisas referentes às consultas de qualquer um dos médicos.
- RF04 – O sistema deve gerar, no final da semana, um relatório contendo quantas consultas foram efetivadas para cada um dos médicos, bem como os planos de saúde.

o  
Ver anotações

Também devem ser definidos os requisitos não funcionais, que estão relacionados a restrições como tempo, imposições de padronizações, entre outros. Existem métricas para se especificar os requisitos não funcionais, o que leva a incluir e determinar as métricas de qualidade.

Tabela 3.1 | Métricas para especificar requisitos não funcionais

Propriedade	Métrica
Velocidade	Transações processadas/segundo. Tempo de resposta do usuário/evento. Tempo de atualização da tela.
Tamanho	Megabytes/número de chips de ROM.
Facilidade de uso	Tempo de treinamento. Número de quadros de ajuda.
Confiabilidade	Tempo médio até a falha. Probabilidade de indisponibilidade. Taxa de ocorrência de faltas. Disponibilidade.
Robustez	Tempo para reiniciar após a falha. Porcentagem de ventos, causando falhas. Probabilidade de corromper dados em uma falha.
Portabilidade	Porcentagem de declarações dependentes do sistema-alvo. Número de sistemas-alvo.

Fonte: Sommerville (2018, p. 94).

Os requisitos não funcionais pensando no sistema do exemplo acima, temos:

- RNF01 – Validação da entrada de dados do paciente.
- RNF02 – Apenas usuários autorizados têm acesso às respectivas funcionalidades.
- RNF03 – Tempo de criação dos relatórios deve ocorrer em <=30segundos.

Esse processo está ligado diretamente com a questão da qualidade, principalmente no que se refere aos objetivos do sistema, portanto, o momento de análise por meio do UML permite que os desenvolvedores busquem ser objetivos e claros a partir de pequenos textos e desenhos de diagramas do sistema a ser desenvolvido, sendo utilizado por toda a equipe de desenvolvimento e pelo cliente. A ferramenta é considerada bastante eficiente no que diz respeito à comunicação entre os membros da equipe de desenvolvimento de sistemas.

Ver anotações

### ■ PLANEJAMENTO DE GESTÃO DE QUALIDADE

Para se definir um planejamento de gestão de qualidade, executar e monitorar, é preciso entender os objetivos do projeto, ou seja, conhecer os requisitos (conforme apresentado no exemplo da metodologia UML), as normas e as diretrizes em que o escopo do projeto está sujeito. A partir desse entendimento, é possível definir os requisitos e as especificações que atenderão à qualidade do produto a ser desenvolvido, atestando, assim, a qualidade em busca do efetivo gerenciamento (CARVALHO, 2018).

Além do guia PMBOK, é preciso utilizar outros processos das áreas de conhecimento, como o acesso ao termo de abertura e o plano de projeto, em que também temos as informações dos custos estimados e do prazo para o encerramento de cada atividade. Com essas informações, é possível identificar os requisitos, as respectivas datas de entregas e definir a qualidade dessas atividades, visando a atender aos padrões definidos pela equipe (COSTA; PEREIRA, 2019).

A escrita do plano de qualidade deve ser mantida com descrições curtas; caso isso não ocorra, provavelmente, as pessoas deixarão de ler o documento e, evidentemente, o propósito desse processo será perdido (SOMERVILLE, 2018).

Após compreender o planejamento do gerenciamento da qualidade integrado às demais áreas do conhecimento, é preciso buscar ferramentas para a análise por meio dos modelos de qualidade, como a análise de custo-benefício, fluxograma, análise de lacunas entre expectativa e percepção, entre outros métodos existentes, bem como escolher o mais adequado para a equipe e a organização (PMI, 2017).

A partir da utilização de modelos de qualidade, temos alguns produtos resultantes do planejamento da qualidade, como: o plano geral de gerenciamento da qualidade; *checklist* que proporciona ao profissional a entrega dos requisitos; planos de melhorias que evitam que processos saiam do controle; e atualizações que ajudam no gerenciamento das comunicações das partes interessadas (CARVALHO, 2018).

As métricas de qualidade definem os requisitos de sucesso do projeto, apontando medidas reais de como a entrega será considerada concluída. Elas fazem parte do plano de gerenciamento da qualidade e devem descrever com detalhes os atributos/as funcionalidade do produto ou processo, ou seja, quando uma tarefa é considerada concluída e quem será o *stakeholder* responsável pela sua aprovação. Os padrões e critérios de aceitação foram predeterminados para cada requisito no processo inicial (COSTA; PEREIRA, 2019).

o  
Ver anotações

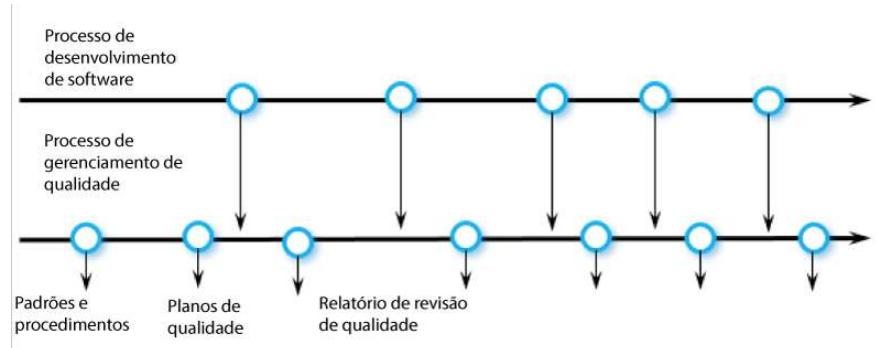
### GARANTIA DA QUALIDADE DO PROJETO

Na etapa seguinte, temos a garantia da qualidade do projeto, em que ocorrem as auditorias internas e/ou externas com as revisões de qualidade, esse processo visa à melhoria contínua. O último item a ser pensado diz respeito ao controle da qualidade do projeto, em que são descritos os procedimentos de inspeção para que o monitoramento ocorra conforme o planejamento desse ciclo de vida do projeto (COSTA; PEREIRA, 2019).

Buscar a excelência do sistema de qualidade de software por meio de padronizações é um caminho proposto desde a Segunda Guerra Mundial e foi potencializado com os métodos Ford para criar padronizações na produção e conseguir avaliar, ao final da produção, se o produto possui a qualidade necessária para ser ofertado ao cliente. "ISO", em grego, significa igual; já em inglês, significa *International Organization for Standardization* (Organização Internacional para a Padronização).

Baseadas nesses conceitos, foram criadas formas de se gerenciar o desenvolvimento de software. A equipe de gerenciamento da qualidade deve ser independente da equipe do processo de desenvolvimento.

Figura 3.4 | Linha dos processos de desenvolvimento de software e de gerenciamento da qualidade



Fonte: Somerville (2018 p. 664).

A equipe de qualidade verificará os resultados a fim de garantir que estejam conforme os padrões e as metas da organização. Essa verificação ocorrerá por meio de documentos, a fim de se verificar se as tarefas relevantes formam todas concluídas ou se ocorreu alguma suposição incorreta referente às atividades de outro grupo.

A gestão da qualidade de software visa a garantir que um software possua poucos defeitos, atingindo um padrão aceitável e pré-acordado de manutenção, confiabilidade, segurança e todos os requisitos que definem qualidade em sistemas de software (SOMMERVILLE, 2018; NEWTON, 2011).

o  
Ver anotações

Essas etapas descritas servem para todo e qualquer projeto de forma genérica, mas quando falamos de desenvolvimento de software, devemos acrescentar a qualidade subjetiva, que está baseada nas características não funcionais, que refletem na experiência do usuário. Caso a funcionalidade não atenda às suas necessidades, arrumará meios de contornar a deficiência e atingir seu objetivo. Porém, se o software não for eficiente ou mesmo confiável, isso fará com que o usuário não alcance seus objetivos.

Portanto, não basta saber se a funcionalidade está implementada e funcionando, é necessário que os atributos não funcionais atendam às expectativas do cliente.

Tabela 3.2 | Atributos não funcionais da qualidade de software

Segurança ( <i>safety</i> )	Compreensibilidade	Portabilidade
Segurança da informação ( <i>security</i> )	Testabilidade	Usabilidade
Confiabilidade	Adaptabilidade	Reusabilidade
Resiliência	Modularidade	Eficiência
Robustez	Complexidade	Apreensibilidade

Fonte: Somerville (2018, p. 667).

Infelizmente, é inviável que se consiga atender plenamente a todos esses atributos não funcionais, por exemplo: a perda de desempenho poderá ser reduzida caso aumente de mais o sistema de segurança. No caso do plano de qualidade para o desenvolvimento de software, devemos incluir a definição do processo de avaliação da qualidade mediante a escolha do atributo de maior relevância, portanto, se a escolha se der pelo desempenho, outros fatores poderão ser sacrificados em prol da escolha.

Outra questão a ser analisada é o desenvolvimento que se dá por meio de um processo criativo, o que significa que a experiência e as habilidades individuais influenciam diretamente nos resultados, o que, fatalmente, poderá sofrer influência por fatores externos, como a pressão para o encerramento do projeto ou o lançamento de novos produtos no mercado. Embora a base da qualidade se apoie em padrões e processos, os gerentes de qualidade se destacam ao reconhecer a existência dos aspectos intangíveis, como a elegância, legibilidade, confiabilidade, manutenibilidade etc., que não conseguem ser inseridas aos padrões. Devendo ser estimuladas por meio do comportamento profissional da equipe em todos os membros (SOMMERVILLE, 2018).

Considerando tudo o que foi definido até aqui, é preciso garantir a qualidade, monitorar e controlar a execução desse plano, de forma a assegurar o andamento das atividades até o seu encerramento, contribuindo com a minimização dos defeitos de planejamento ou eliminando-os nas inspeções, durante o andamento do projeto (COSTA; PEREIRA, 2019).

o  
Ver anotações

As inspeções de programas ocorrem por meio das revisões feitas por pares, em que os bugs são encontrados pelos próprios integrantes da equipe, fazendo, assim, parte do processo de verificação e validação de software, os sistemas em desenvolvimento contam com a possibilidade de verificação, utilizando as representações dos modelos UML. Nas reuniões de inspeções, os membros que realizaram a revisão linha a linha do código-fonte os descrevem para a equipe. Para esta atividade é recomendado o uso de um checklist, podendo ser criado pela equipe ou utilizando modelos (templates) já sugeridos, estes itens irão sofrer variações conforme a linguagem de desenvolvimento (SOMERVILLE, 2018).

## MÉTODOS DE MEDIÇÃO DE QUALIDADE

Uma das grandes dificuldades da gestão da qualidade de desenvolvimento de software são os métodos de medição de qualidade, afinal o software não sofre desgaste. A ISO/IEC 25010, de 2011, propõe um modelo que define oito categorias de características sobre a qualidade, sendo estas divididas em subcaracterísticas, conforme Quadro 3.10.

Quadro 3.10 | Modelo de qualidade de produto de sistemas proposto pela Organização Internacional para Padronização e pela IEC - ISO/IEC 25010, que se relacionam com propriedades estáticas do software e propriedades dinâmicas do sistema de computador

Características	Subcaracterísticas	Significado
Adequação funcional: satisfaz as necessidades explícitas e implícitas para se atingir a finalidade do produto?	Completude funcional	É adequado às necessidades do usuário?
	Correção	Faz o que é proposto de acordo com a necessidade?
	Adequação funcional	As funções realizam as tarefas de forma objetiva e fácil?
Eficiência: os recursos e os tempos utilizados são compatíveis com o nível de desempenho requerido para o produto?	Comportamento do tempo	O tempo de resposta e de processamento atende aos requisitos?
	Utilização de recursos	Quanto recurso utiliza? Durante quanto tempo?
	Capacidade	Os limites máximos atendem aos requisitos?

Características	Subcaracterísticas	Significado
Compatibilidade: a troca de informações entre produtos, sistemas, componentes e/ou execução das suas funcionalidades ao compartilhar o mesmo ambiente de hardware ou de software.	Coexistência	As funcionalidades são eficientes ao compartilhar o ambiente comum e os recursos com outros produtos? Existe o impacto com algum outro produto?
	Interoperabilidade	Dois ou mais sistemas, produto ou componentes trocam informações e as usam com eficiência e segurança?
Usabilidade: é fácil de usar o software com eficiência, eficácia e satisfação em um contexto de uso específico?	Reconhecimento de adequação	O usuário reconhece que o produto ou sistema é adequado as suas necessidades?
	Capacidade de aprendizagem	O usuário consegue aprender a usar o produto ou sistema com eficácia, eficiência, isenção de risco e satisfação em um contexto de uso especificado?
	Operabilidade	É fácil operar e controlar?
Estética da interface do usuário	Proteção ao erro do usuário	O sistema evita que o usuário cometa erros?
	Acessibilidade	O sistema pode ser usado por pessoas com a mais ampla gama de características e capacidades, (deficiências) para atingir uma meta especificada?

Ver anotações

Características	Subcaracterísticas	Significado
Confiabilidade: durante um período, continua funcionando de acordo com as condições preestabelecidas?	Maturidade	Atende às necessidades de confiabilidade sob operação normal?
	Disponibilidade	O tempo em que o sistema está operacional e acessível para uso?
	Tolerância a erro	Apesar das falhas de hardware ou software, o sistema permanece funcionando?
	Recuperabilidade	É capaz de recuperar dados após falha e restabelecer o sistema?
Segurança: o sistema protege as informações e os dados, fornecendo o grau de acesso adequado para seu tipo e nível de autorização.	Confidencialidade	Existe a garantia de que os dados estão restritos a determinados usuários?
	Integridade	O sistema impede o acesso não autorizado ou modificações no código?
	Não repúdio	Existe um controle das atividades dos usuários de forma que eventos ou ações não sejam repudiados posteriormente?
	Prestação de contas	As ações de uma entidade podem ser rastreadas exclusivamente para essa entidade?
	Autenticidade	Há garantia de que o usuário ou recurso pode ser comprovado?

Ver anotações

Características	Subcaracterísticas	Significado
Manutenibilidade: há eficácia e eficiência para correções, atualizações e alterações?	Modularidade	A mudança em um componente tem impacto mínimo em outros componentes?
	Reutilização	Um ativo pode ser usado em mais de um sistema ou na construção de outros ativos?
	Analisabilidade	É eficaz a avaliação do impacto de uma alteração pretendida ou o diagnóstico de uma falha quando ocorre? E a modificação?
	Modificabilidade	É eficaz e eficiente modificar sem inserir defeitos ou degradar a qualidade do produto existente?
	Testabilidade	Há eficácia e eficiência nos critérios de testes realizados para se determinar se esses critérios foram atendidos?
Portabilidade: é utilizável o produto em plataformas diferentes com pequeno esforço de adaptação?	Adaptabilidade	É fácil adaptar a outros ambientes sem aplicar outras ações ou meios além dos fornecidos para essa finalidade no software?
	Capacidade para ser instalado	É eficaz e eficiente instalar e desinstalar em ambiente específico?
	Capacidade para substituir	É fácil substituir por outro software que possui a mesma finalidade ao atualizar?

Fonte: adaptado de ABNT (2011).

Foi apresentado apenas o modelo de qualidade do produto; falamos de uma única normativa de um conjunto sugerido pelo ISO/IEC 25010, portanto, como se observar no PMBOK, é importante o devido aprofundamento para cada necessidade, e sua implementação deve ocorrer de forma parcial ou completa e adequada a cada realidade.

o  
Ver anotações

## CONTROLE DA QUALIDADE

No processo de controle da qualidade é onde ocorre o registro dos resultados da execução das atividades, a avaliação do desempenho e as recomendações necessárias para a melhoria contínua. Para essa etapa, os resultados que podem ser obtidos são: as validações das mudanças, as informações referentes ao desempenho do desenvolvimento, os documentos com as solicitações de mudanças, a atualização dos documentos conforme necessidade e a verificação das entregas. Portanto, esse processo de controle de qualidade deve ocorrer nas fases de execução e finalização, para que se possa mostrar ao cliente que todos os critérios definidos inicialmente foram cumpridos (COSTA; PEREIRA, 2019).

Para uma ilustração desse processo de encerramento, vamos observar um documento com solicitação de mudanças e os resultados obtidos com ele.

Quadro 3.11 | Sugestão básica para relatório de controle de mudança com versionamento visando ao controle da qualidade

Controle de Versões			
Versão	Data	Autor	Notas da Revisão
1.0	27/05/2020	Sr. XXX (sponsor)	Elaboração Inicial – primeiro rascunho.
1.1	10/06/2020	Sr. XXX (sponsor)	Revisão pós-reunião com fornecedores e responsável pelo setor de compras.
2.0	31/03/2012	Sr. XXX (sponsor)	Revisão final.
2.1	31/03/2013	Sr. XXX (sponsor)	Inclusão de mais 2 indicadores IQ02, IQ03.
2.2	31/05/2015	Sr. XXX (sponsor)	Migrado para o plano de gerenciamento da qualidade.
2.3	29/06/2015	Sr. XXX (sponsor)	Revisão do programa de melhoria contínua.

Nº Solicitação	SM02	Solicitantes	Sr XXX (sponsor); Sr XXX (Responsável pelo setor).
Prioridade [0-Maior prioridade. 5-Menor]	3	Setor solicitante	Compras

<b>Descrição da mudança</b>
Na criação do relatório de solicitação de orçamento por período e fornecedores, incluir o status de resposta e atendimento adequado aos produtos adquiridos referente ao cumprimento de prazo, custo e qualidade.

<b>Justificativa</b>
Relatório no âmbito de maior complexidade de análise não previsto inicialmente. Sua necessidade surgiu a partir da implementação da gestão total de qualidade em processo de implantação na empresa. Após a inserção da funcionalidade de avaliação dos fornecedores, quanto ao cumprimento dos requisitos de atendimento as compras efetuadas.

<b>Classificação de impacto no projeto</b>								
<table border="1"> <thead> <tr> <th>Análise de Impacto</th> <th>Descrição</th> </tr> </thead> <tbody> <tr> <td>Esforço Estimado (Horas)</td> <td>16 horas de um desenvolvedor; 4 horas de outro desenvolvedor para teste.</td> </tr> <tr> <td>Custo Estimado (R\$)</td> <td>R\$ 3.000,00.</td> </tr> <tr> <td>Impacto no Prazo (Dias)</td> <td>Mais três dias.</td> </tr> </tbody> </table>	Análise de Impacto	Descrição	Esforço Estimado (Horas)	16 horas de um desenvolvedor; 4 horas de outro desenvolvedor para teste.	Custo Estimado (R\$)	R\$ 3.000,00.	Impacto no Prazo (Dias)	Mais três dias.
Análise de Impacto	Descrição							
Esforço Estimado (Horas)	16 horas de um desenvolvedor; 4 horas de outro desenvolvedor para teste.							
Custo Estimado (R\$)	R\$ 3.000,00.							
Impacto no Prazo (Dias)	Mais três dias.							

Requisito de Qualidade	Ações para atingimento	Indicadores
Tempo de resposta do relatório <= 30seg	Verificação do código de desenvolvimento; Sistema de gerenciamento de banco de dados adequado; Tempo de resposta da estrutura do servidor.	IQ01 – revisão em pares. IQ02 – testes pela equipe de teste. IQ03 – auditoria. IQ04 – teste com os usuários.
Entrega dentro do prazo	Comprometimento da equipe.	
Criação da documentação	A equipe de desenvolvimento e testes deve ter fácil acesso aos documentos compartilhados.	
Manutenibilidade	Código comentado. Documentos atualizados.	
Definição de pronto	Alinhar o documento que define quando a funcionalidade é considerada concluída.	

Aprovações		
Participante	Assinatura	Data
Patrocinador do projeto		
Gerente do projeto		

Fonte: adaptado do PMI (2017).

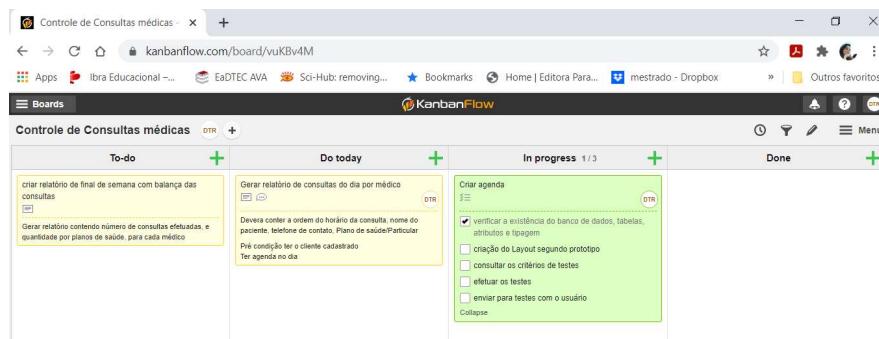
Existe uma definição de pronto (*Definition of Done*, no original, ou *DoD*) para a entrega de cada atividade que é tratada pelo método *Scrum* como uma criação desenvolvida pela equipe durante a *sprint* (período em que a equipe desenvolve uma parte do produto final), tendo que ser submetida a todas as etapas do desenvolvimento e validação, a fim de que possa ser dada como pronta (DONE). Esse processo de verificação reforça a transparência entre os integrantes da equipe, já que é o principal pilar da metodologia *Scrum*, tornando público o conhecimento do significado DONE. Para que exista uma definição de pronto bem elaborada, é necessário o envolvimento do Time *Scrum* por completo, a fim de que sejam definidos os aspectos e a construção do produto, de testes e homologação de cada um dos itens do *backlog* da *sprint* (DUARTE, 2019).

Para ilustrar um exemplo de DoD, podemos ter: codificação, revisão do código, aprovação nos testes individuais, ser aprovado nos testes de funcionalidades manuais e/ou automatizados, versionamento e, por fim, a homologação que, em geral, dá-se pelo *Product Owner*, seu cliente no método *Scrum*.

Pensando nesse processo, é possível utilizar o método Kanban, com 3 colunas divididas em *TO DO* (fazer), *Work in Progress* (WIP – desenvolvendo) e *DONE* (pronto).

A fim de se ampliar o conceito de pronto para que haja uma boa definição, é preciso pensar em critérios de qualidade e incluir os requisitos não funcionais, como já discutido anteriormente na checklist proposta pela norma ISO 25010, no final do desenvolvimento é fundamental que se tenha passado por todas as análises e testes previamente definidos. Como resultado desse processo, a cada sprint trabalhada, haverá uma evolução, o que trará o aumento da qualidade no desenvolvimento do sistema, fazendo com que o empirismo inerente a cada ciclo gere a melhoria contínua, produzindo, assim, softwares com maior qualidade (DUARTE, 2019).

Figura 3.5 | Exemplo de quadro de KanBan durante o desenvolvimento de um sistema de controle administrativo de consultas médicas



Fonte: adaptada de Costa e Pereira (2019).

Para a utilização desse quadro, basta acessar o site e fazer o cadastro no sistema web. Um quadro pronto aparecerá para que você possa editá-lo, e já será apresentado um breve tutorial interativo de como funciona a ferramenta.

Modifique o nome do quadro para um nome adequado com o projeto. Para cada

o

Ver anotações

tarefa incluída, poderão ser definidos os responsáveis, o tempo estimado e o tempo realizado, um detalhamento da tarefa, bem como inserir a descrição das tarefas, principalmente no que se refere ao processo de definição de pronto e métricas de qualidade.

Existem inúmeras outras ferramentas gratuitas para a utilização do quadro de Kanban, como o site Padlet.

Com todo esse conhecimento referente à qualidade no desenvolvimento de sistemas, incluindo o que já foi aprendido anteriormente sobre como mitigar riscos, ou seja, gestar todos esses quesitos, possuir conhecimento das técnicas de gestão de projeto, conhecer os cinco grupos de processo de gestão com normalizações reconhecidas nacionalmente e internacionalmente, saber escolher como será gestado o seu projeto mediante as características da equipe e da organização envolvida e tento algumas ferramenta para auxiliar nesses desafios do mundo de desenvolvimento de software, entendemos que, neste momento, você passa a ser um profissional capaz de resolver problemas e analisa-los cuidadosamente, planejar e organizar as atividades, bem como promover um relacionamento interpessoal saudável, com empatia e ética no trabalho, sabendo gerir os conflitos em todos os níveis e ser criativo e inovador em todos os processos, atendendo às necessidades de adaptação e, portanto, sendo flexível conforme a circunstância.

#### FAÇA VALER A PENA

##### Questão 1

Para alcançar uma compatibilidade com a qualidade, é importante reconhecer a gestão da qualidade por meio de aspectos diferentes. Esses aspectos possuem pontos de vista distintos, a depender diretamente do atendimento ou não às expectativas do *stakeholder*.

- I. A Satisfação do cliente está no atendimento de suas expectativas.
- II. A gestão da empresa possui sua responsabilidade quanto a garantir a qualidade.
- III. A qualidade do produto está desvinculada dos interesses organizacionais como o custo, o prazo e o escopo.

Considerando o contexto apresentado, assinale a alternativa correta.

a. Apenas as afirmativas I e II estão corretas.

b. Apenas a afirmativa I está correta.

c. Apenas a afirmativa II está correta.

d. Apenas as afirmativas I e III estão corretas.

e. As afirmativas I, II e III estão corretas.

##### Questão 2

Uma das grandes dificuldades da gestão da qualidade de desenvolvimento de software são os métodos de medição de qualidade, afinal, o software não sofre desgaste. A ISO/IEC 25010, de 2011, propõe um modelo que define oito categorias

o

Ver anotações

de características sobre a qualidade, sendo elas divididas em subcaracterísticas.

Característica	Exemplo
I. Adequação funcional	a. O sistema compartilha informações com os demais sistemas ao compartilhar o mesmo hardware ou software.
II. Eficiência	b. O funcionamento se dá de acordo com o que foi estabelecido por um longo período de tempo.
III. Compatibilidade	c. Facilidade no processo de correções e manutenção do sistema.
IV. Usabilidade	d. O desempenho requerido pelo produto é atendido no que diz respeito a tempo e recurso.
V. Confiabilidade	e. Cada usuário tem o seu nível de acesso definido.
VI. Segurança	f. Com pouca adaptação, é possível utilizar em SO diversos.
VII. Manutenibilidade	g. O sistema atende a todas as necessidades implícitas do software.
VIII. Portabilidade	h. Dentro de um contexto de uso, o software possui eficiência, eficácia e satisfação.

Assinale a alternativa que apresenta a associação CORRETA entre as colunas.

a. I-a; II-h; III-g; IV-d; V-e; VI-c; VII-f; VIII-b.

b. I-d; II-a; III-h; IV-b; V-e; VI-c; VII-f; VIII-a.

c. I-f; II-g; III-d; IV-a; V-h; VI-b; VII-e; VIII-c.

d. I-g; II-d; III-a; IV-h; V-b; VI-e; VII-c; VIII-f.

e. I-c; II-f; III-g; IV-d; V-a; VI-h; VII-b; VIII-e..

### Questão 3

Em um projeto de desenvolvimento de sistema, visando à qualidade do sistema a ser desenvolvido, utilizando-se da metodologia Scrum, é relevante a existência de critérios de avaliação de cada uma das entregas a ser efetivada. Porém, para que essa tarefa seja considerada concluída, a equipe precisa ter a compreensão do significado de pronto (DoD). Esse processo é relevante para que não exista dúvidas sobre quando o desenvolvedor pode dizer "ACABEI!"

Tomando como referência o contexto apresentado acima, julgue cada uma das afirmativas a seguir como (V) verdadeira ou (F) falsa.

( ) A equipe Scrum cria os critérios de DoD para cada *sprint* no início da *sprint*.

( ) O processo de DoD reforça a transparência entre os integrantes, o principal pilar do *Scrum*.

( ) O envolvimento de toda a equipe para a definição dos aspectos de construção, testes e homologação torna a definição de pronto melhor elaborada.

Assinale a alternativa que apresenta a sequência CORRETA.

a. V – F – V.

b. F – V – V.

c. V – V – F.

d. V – F – V.

e. F – F – V.

Ver anotações

## REFERÊNCIAS

ABNT — Associação Brasileira de Normas Técnicas. **NBR ISO/IEC 9126-1**.

Engenharia de software - Qualidade de produto - Parte 1: Modelo de qualidade. Rio de Janeiro: ABNT, 2003.

CARVALHO, F. C. A. (org.). **Gestão de projetos**. São Paulo: Pearson Education do Brasil, 2015.

CARVALHO, F. C. A de. **Gestão de projetos**. 2. ed. Pearson Education do Brasil, 2018.

COSTA, A. B. da; PEREIRA, F. da S. **Fundamentos de gestão de projetos**: da teoria à prática – como gerenciar projetos de sucesso. Curitiba: Intersabers, 2019.

DUARTE, L. F. **Agile coaching**: um guia prático. 2. ed. [S.I.]: eBook Kindle, 2019.

ISO — International Organization for Standardization. **ISO/IEC 25010**. System and Software engineering - System and software Quality Requirements and Evaluation (SQuaRE) - system and software quality models. Geneva: ISO, 2011.

JALOTE, P. **An integrated approach to software engineering**. 3. ed. New York: Springer, 2005.

PMI — Project Management Institut. **Guia do conhecimento em gerenciamento de projetos**: guia PMBOK. 6. ed. [S.I.]: PMI, 2017.

PRESSMAN, R. S. **Engenharia de software**. 6. ed. Rio de Janeiro: McGraw-Hill, 2006. 720p.

ROCHA, A. R. C.; MALDONADO, J. C.; WEBER, K. C. **Qualidade de software: teoria e prática**. São Paulo: Prenttice Hall, 2001. 303p.

SOMMERVILLE, I. **Engenharia de software**. 10. ed. São Paulo: Pearson Prentice Hall, 2018.

WEBER, K. C. Tecnologia da informação: programa brasileiro da qualidade e produtividade em software. [S.I.: s.n.], 2008. 485 p.