



APRENDIZAGEM EM FOCO

GERENCIAMENTO ÁGIL DOS SISTEMAS



APRESENTAÇÃO DA DISCIPLINA

Autoria: Marco Ikuro Hisatomi

Leitura crítica: Valéria Cristina Gomes Leal

Aluno, você vai aprimorar seus conhecimentos em gerenciamento ágil a partir das orientações desta disciplina em função de:

1. Conhecer metodologia que valoriza o domínio das regras de negócio e obtenção do comprometimento do cliente com o processo de desenvolvimento e manutenção do sistema.
2. Conhecer técnicas de métricas e estimativas para gerenciamento de escopo, riscos, custo e qualidade.
3. Conhecer os submodelos em estimativa de custos Cocomo II.
4. Conhecer a estratégia para a implantação da gestão da qualidade para o processo de desenvolvimento e manutenção do software.
5. Compreender como os processos de verificação e validação são úteis no gerenciamento ágil dos sistemas.
6. Saber reconhecer qual é a consequência em qualidade quando se classifica um erro, defeito e falha.
7. Entender como classificar uma manutenção de correção ou de evolução e os efeitos na administração na alteração do software.
8. Entender como a ferramenta de gerenciamento de configuração auxiliará engenheiros de software e gestores no controle de código-fonte e versão do produto.
9. Aprender a elaborar uma proposta de manutenção durante a administração de sistemas, com capacitação para análise de indicadores de gestão das alterações do software.
10. Debater sobre a gestão da qualidade no processo de desenvolvimento de sistemas, quanto aos fatores da qualidade e maturidade no processo.

11. Assistir aos vídeos, da leitura de conteúdo, da pesquisa em acervo bibliográfico disponíveis em biblioteca on-line, ouvir *podcast* e realizar exercícios.
12. Compreender os papéis fundamentais no processo de desenvolvimento entre engenheiro de software, usuários, cliente e gestor do projeto.

INTRODUÇÃO

Olá, aluno (a)! A *Aprendizagem em Foco* visa destacar, de maneira direta e assertiva, os principais conceitos inerentes à temática abordada na disciplina. Além disso, também pretende provocar reflexões que estimulem a aplicação da teoria na prática profissional. Vem conosco!

TEMA 1

Controlando o escopo: as métricas e as estimativas

Autoria: Marco Ikuro Hisatomi

Leitura crítica: Valéria Cristina Gomes Leal





DIRETO AO PONTO

Existem muitas técnicas e recursos que podem ser utilizados para que se consiga o controle do escopo num projeto de desenvolvimento de sistemas. Contudo, em métodos ágeis, a estimativa é um dos meios mais eficientes para controlar o escopo de um projeto de software. Ainda que seja impreciso medir as funcionalidades (as histórias de usuário), por estarem em constante mudança, além de ser um “objeto” subjetivo.

O grande trunfo está, exatamente, em gerenciar, dentro de um espaço limitado de tempo, o que pode ser desenvolvido seguindo as necessidades reais do cliente e dos seus negócios. Por sua vez, a métrica é aplicada para impulsionar a velocidade e a qualidade do produto entregue, inclusive, aproveitando a capacidade produtiva da equipe em desenvolver o que está sendo definido pelo cliente.

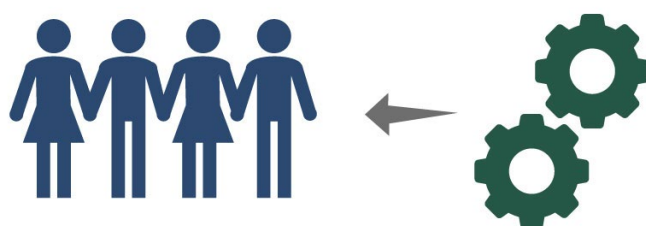
Todavia, manter uma estrutura envolvendo pessoas, o principal recurso de um projeto de software, só se torna possível por seguir valores e princípios estabelecidos pelo manifesto ágil, que dão sustentação no gerenciamento das mudanças constantes do escopo.

Agora, entenda quais são os quatro valores, base fundamental do manifesto ágil, para o gerenciamento de atividades de um projeto.

- I. Pessoas interagindo para esclarecer e manter o escopo atualizado é mais importante do que saber utilizar ferramentas e técnicas do desenvolvimento de software. Uma organização produzirá software com pessoas muito mais do que se utilizasse somente processos e ferramentas.

A cada momento de interação entre equipe e usuários, a comunicação se torna mais eficiente (buscando melhoria no entendimento), mais eficaz (focando nos objetivos) e diminuindo a distância entre todos os envolvidos (*stakeholders*).

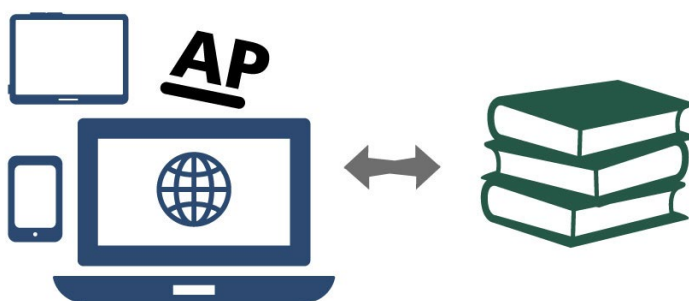
Figura 1 – Indivíduos e interações acima de processos e ferramentas



Fonte: elaborada pelo autor.

- II. A produção de documentação é importante, desde que se faça uso e agregue valor. O software funcionando é muito mais importante e os clientes querem resultados operacionais por meio das funcionalidades implementadas no software, não pela documentação.

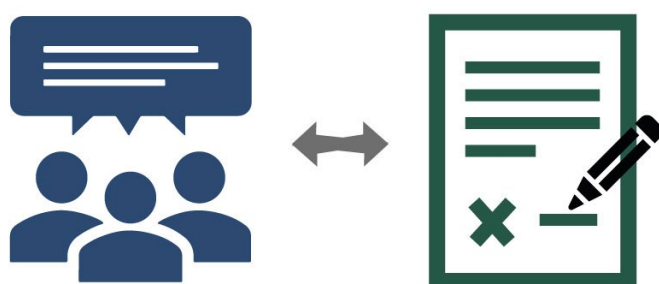
Figura 2 – Software funcionando acima de documentações



Fonte: elaborada pelo autor.

III. A colaboração entre clientes e equipe é acima de qualquer contrato firmado em papel, pois tudo vai ser decidido rapidamente no trabalho em conjunto da equipe com o usuário final. A busca de um só objetivo deve ser o resultado da colaboração entre as partes.

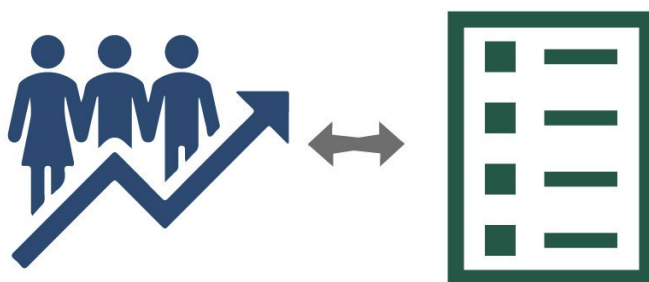
Figura 3 – Colaboração entre equipe e cliente acima de contrato



Fonte: elaborada pelo autor.

IV. Para acompanhar a demanda de novas necessidades do mercado, com incertezas constantes, a atenção deve estar direcionada a essas mudanças, mais que seguir um plano que pode estar obsoleto.

Figura 4 – Mudanças em negócios acima de plano estabelecido



Fonte: elaborada pelo autor.

Enfim, o “Direto ao ponto” deste tema é: procurar o detalhamento dos requisitos e das regras de negócio no prazo mais curto possível da entrega do software funcionando, focado nas necessidades do negócio do usuário final, por meio da interação constante entre equipe do projeto e o usuário final.

Referências bibliográficas

PRESSMAN, R. S. **Engenharia de software**: uma abordagem profissional. Porto Alegre: Amgh, 2016.


SOMMERVILLE, I. **Engenharia de software**. São Paulo: Pearson Education do Brasil, 2018.

PARA SABER MAIS

O movimento ágil foi se tornando mais visível pelos desenvolvedores de software a partir da criação de métodos ágeis, sendo um dos mais conhecidos o Scrum, este que é um *framework* para desenvolver, entregar e manter produtos complexos (SCHWABER; SUTHERLAND, 2017).

Por prever procedimentos simples, conexos ao Manifesto Ágil, tornou-se comum a sua adoção em desenvolvimento de softwares por demonstrar eficácia para projetos que exigem complexidade, requisitos em constante mudança e prazos curtos (PRESSMAN, 2016).

Você que estará envolvido no ambiente dinâmico do desenvolvimento de software, provavelmente vai se deparar com práticas do Scrum (mesmo que parcialmente). Ou, pelo menos, vai perceber que faz muito sentido se praticar o que preconiza no



Guia do Scrum. Portanto, é recomendável ter habilidades nesse *framework*, pois é provável que tenha oportunidade de fazer parte de um time Scrum.

A partir da afirmação de que “o conhecimento vem da experiência e de tomada de decisões baseadas no que é conhecido. O Scrum emprega uma abordagem iterativa e incremental para aperfeiçoar a previsibilidade e o controle de riscos”, segundo o *Guia do Scrum* (SCHWABER; SUTHERLAND, 2017, p. 4).

Uma das características fundamentais do Scrum é que os eventos são *time boxed*, de tal modo que todo evento tem uma duração máxima, complementando essa característica.

[...] cada evento no Scrum é uma oportunidade de inspecionar e adaptar alguma coisa. Estes eventos são especificamente projetados para permitir uma transparência e inspeção criteriosa. Falhas na inclusão de qualquer um destes eventos resultará na redução da transparência e na perda de oportunidades para inspecionar e adaptar. (SCHWABER; SUTHERLAND, 2017, p. 4.)

A prática da comunicação constante, observada pelos métodos ágeis, no Scrum, acontece a reunião diária, um evento *time boxed* de 15 minutos, que tem como proposta, além de inspecionar o progresso do projeto, identificar e remover impedimentos para o desenvolvimento, entre outros benefícios fundamentais numa metodologia que defende a transparência, inspeção e adaptação.

Referências bibliográficas

PRESSMAN, R. S. **Engenharia de software**: uma abordagem profissional. Porto Alegre: Amgh, 2016.

SCHWABER, K.; SUTHERLAND, J. **Guia do Scrum** – Um guia definitivo para o Scrum: as regras do jogo. Share-Alike da Creative Commons. 2017. Disponível em: <https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-Portuguese-Brazilian.pdf>. Acesso em: 20 mai. 2020.

TEORIA EM PRÁTICA

Imagine uma história do usuário assim: finalizar a venda de fotografia pela web, onde o cliente escolhe uma fotografia, o tamanho da imagem, um endereço de entrega (diferente do já existente no cadastro) e efetua o pagamento via cartão de crédito. Na fase de planejamento da iteração, a equipe percebe que a estimativa em pontos da história está muito grande para conseguir entregar numa única iteração. Reflita sobre esta situação para conseguir realizar a entrega adequada. Deve aumentar o tempo da iteração? Aumentar a equipe de desenvolvimento? Diminuir o tempo de testes? Deixar de efetuar a refatoração?

Para conhecer a resolução comentada proposta pelo professor, acesse a videoaula deste *Teoria em Prática* no ambiente de aprendizagem.

LEITURA FUNDAMENTAL

Indicações de leitura

Indicação 1

O texto proposto para leitura tem como objetivo a sua preparação como desenvolvedor de software: sob os princípios do processo

ágil. Isso significa que deverá estar preparado para mudanças rápidas, manter o foco nos objetivos dos usuários finais, que deverá estar atualizado constantemente, entregar o software funcionando e com adaptação incremental, e controlar o escopo dos sistemas.

Para realizar a leitura, acesse a plataforma Biblioteca Virtual da Kroton e busque pelo título da obra no parceiro Minha Biblioteca.

PRESSMAN, R. S. **Engenharia de software**: uma abordagem profissional. Porto Alegre: Amgh, 2016. p. 69-72.


Indicação 2

Se a história do usuário é a principal fonte para o desenvolvimento de software e a partir dela será controlado o escopo dos sistemas, então é muito importante que você tenha o domínio do conhecimento sobre esse assunto. Vai perceber que, para entender uma história escrita pelo usuário, é preciso vivenciar com frequência o dia a dia operacional dos negócios dele. Nem sempre apenas a descrição estará suficientemente completa para contemplar as funcionalidades e os comportamentos do software.

Convido a ler o item “História do usuário”, para ficar mais preparado para colaborar com o cliente na concepção de uma história do usuário.

Para realizar a leitura, acesse a plataforma Biblioteca Virtual da Kroton e busque pelo título da obra no parceiro Biblioteca Virtual 3.0_Pearson.

SOMMERVILLE, I. **Engenharia de software**. São Paulo: Pearson Education do Brasil, 2018. p. 63-65.



QUIZ

Prezado aluno, as questões do Quiz têm como propósito a verificação de leitura dos itens *Direto ao Ponto, Para Saber Mais, Teoria em Prática e Leitura Fundamental*, presentes neste *Aprendizagem em Foco*.

Para as avaliações virtuais e presenciais, as questões serão elaboradas a partir de todos os itens do *Aprendizagem em Foco* e dos slides usados para a gravação das videoaulas, além de questões de interpretação com embasamento no cabeçalho da questão.

1. Escolha a alternativa que preenche corretamente a afirmativa:
“As histórias de usuário podem ser utilizadas no ____ das ____ do sistema. Cada cartão de história elaborado pelo time de desenvolvimento possui a decomposição da história do usuário que receberá uma ____ de recursos necessários para a implementação” (SOMMERVILLE, 2018).
 - a. Planejamento; iterações; documentação.
 - b. Orçamento; projeções; estimativa.
 - c. Planejamento; iterações; estimativa.
 - d. Orçamento; criações; codificação.
 - e. Planejamento; criações; documentação.
2. Ao perceber que o tempo da iteração não foi suficiente para a implementação e entrega da história do usuário, deve-se proceder a um ajuste para garantir que o objetivo da iteração seja alcançado. Assinale a alternativa que especifica o ajuste adequado para esse caso.

- a. Aumentar o tempo da iteração.
- b. Contratar novos integrantes para o time de desenvolvimento.
- c. O time de desenvolvimento deve reescrever a história do usuário.
- d. O usuário final deve reescrever a história em histórias menores.
- e. Eliminar parte da história para conseguir entregar o software.



GABARITO

Questão 1 - Resposta C

Resolução: Conforme o texto da página 64 do livro *Engenharia de software* descreve o modelo de processo ágil quanto às histórias de usuário e as estimativas. É a partir das histórias de usuário que se planeja o que será desenvolvido na iteração, sendo que cada história dessa recebe uma estimativa de esforço para o seu desenvolvimento.

Questão 2 - Resposta D

Resolução: O compromisso de escrita da história do usuário é do próprio usuário, inclusive, quando existe a necessidade de divisão em partes menores, pois é ele que tem a responsabilidade da história e da prioridade de desenvolvimento.

TEMA 2

Gestão de riscos e de custos com Cocomo II

Autoria: Marco Ikuro Hisatomi

Leitura crítica: Valéria Cristina Gomes Leal





DIRETO AO PONTO

Para que você possa controlar os riscos em projetos de software, como deve ter percebido, é preciso eliminar as fontes (causas) que possam provocar um impacto negativo ao andamento das atividades do desenvolvimento ou até mesmo a interrupção.

Vejamos, se num determinado momento ocorre uma falha no gerenciador de banco de dados do ambiente de desenvolvimento. Bem, aqui podem ser percebidas algumas consequências: a) pausa nas atividades do desenvolvimento; b) início de atividades não previstas para identificação do problema; c) necessidade de troca ou ajuste do ambiente de desenvolvimento; d) necessidade de troca do sistema gerenciador de banco de dados; entre outras.

Enfim, a partir de uma ocorrência “falha no gerenciador de banco de dados”, uma variedade de consequências ou impactos vão surgir à medida que vão avançando as investigações, mas só será possível estancar e resolver o problema quando a causa fundamental esteja controlada.

Sendo assim, a visão de gerenciamento do processo de software, sob a perspectiva dos riscos, necessariamente, acontecerá, ou pelo planejamento para evitar ou minimizar as causas, ou por inspeção e investigação se não conseguirmos evitar os causadores de falhas no decorrer do processo.

Os eventos considerados críticos podem ser elencados em várias fases do processo de software, desde a fase de iniciação, durante o processo de desenvolvimento e até na fase de administração do sistema que se encontra em operação. Em detrimento das demandas de mercado e de ambiente de desenvolvimento de software, estudos de redução de riscos em projetos ágeis, por

histórias de usuários incompletas ou ambíguas, por necessidade de evolução da tecnologia inovadora do ambiente operacional, por atualização de regras de negócios e requisitos de usuários e por forças das mudanças de recursos humanos, software ou hardware no ambiente de desenvolvimento.

Figura 1 – Eventos causadores de falhas



Fonte: adaptada de Lai (2015, p. 31).

A Figura 1 mostra os principais causadores de mudanças por fator interno, tanto as mudanças de recursos quanto a identificação de problemas em requisitos, porém este é impulsionado por fatores externos. Por outro lado, de fator externo com grande impacto no desenvolvimento de software são a adaptação às novas tecnologias (exemplo: requisição do uso de inteligência artificial) e a atualização de requisitos (exemplo: acrescentar consulta do cadastro nacional para validar o número do Cadastro de Pessoa Física). Embora todas elas são de carácter obrigatório para não se tornar obsoleto e, conseqüentemente, inoperante, assim garantindo a sobrevivência dos sistemas.

Referências bibliográficas

LAI, S. T. Maintainability enhancement procedure for reducing agile software development risk. **International Journal of Software Engineering Applications (IJSEA)**, Vol. 6, N. 4. 2015.

SOMMERVILLE, I. **Engenharia de software**. São Paulo: Pearson Education do Brasil, 2018.

PARA SABER MAIS

Para a composição de custos do desenvolvimento de sistemas, além dos valores relacionados diretamente ao projeto, pessoas do desenvolvimento, recursos computacionais diretamente relacionados à implementação, existem os custos fixos¹, conforme relacionado por Sommerville (2018):

Os custos do esforço não são apenas os salários dos engenheiros de software envolvidos no projeto. As organizações calculam os custos do esforço em termos de custos indiretos, onde assumem o custo total da administração da organização e o dividem pelo número de funcionários produtivos. Portanto, os seguintes custos fazem parte do custo total do esforço:

1. os custos de fornecimento, aquecimento e iluminação de escritórios;
2. os custos de recrutamento, preparação de propostas e marketing;
3. os custos da equipe de suporte, como contadores, administradores, gerentes de sistema, produtos de limpeza e técnicos;
4. os custos de redes e comunicações;

¹ Sommerville elenca os principais custos fixos. Disponível em: <https://iansommerville.com/software-engineering-book/static/web/overhead-costs/>. Acesso em: 30 mai. 2020.

5. os custos de instalações centrais, como uma biblioteca ou instalações recreativas;
6. os custos da Previdência Social e benefícios dos empregados, como pensões e seguro de saúde.

Esse fator de sobrecarga geralmente é pelo menos o dobro do salário do engenheiro de software, dependendo do tamanho da organização e dos custos associados. Portanto, se uma empresa paga a um engenheiro de software US\$ 110.000 por ano, seus custos totais são de pelo menos US\$ 220.000 por ano ou cerca de US\$ 18.000 por mês.

Uma vez que o projeto de desenvolvimento esteja sendo supervisionado sob o aspecto financeiro/monetário, faz-se necessário compor ao custo total do projeto esses custos fixos, ou qualquer outro valor que possa comprometer o sucesso do desenvolvimento.

Referências bibliográficas

SOMMERVILLE, I. **Engenharia de software**. São Paulo: Pearson Education do Brasil, 2018.

TEORIA EM PRÁTICA

Se o seu time de desenvolvimento de um sistema complexo, com necessidade de interação com outras equipes que estão em projetos onde os sistemas se integram; e, ainda, sabendo que manter um custo baixo ao longo do ciclo de vida é um desafio importante, independentemente, utiliza-se o modelo de desenvolvimento tradicional ou métodos ágeis. Faça uma reflexão do comparativo entre um processo de software convencional e

um processo ágil quanto aos custos. Acredita que os custos serão menores usando métodos ágeis? Quando ocorrer as mudanças de requisitos do sistema, qual é a tendência de aumento dos custos? Bem, Pressman (2016 *apud* AMBLER, 2004) argumenta que, um processo ágil, quando aplicado adequadamente, propicia um crescimento menor em custos ao longo do ciclo de vida de um sistema, comparado com os modelos tradicionais .

PRESSMAN, R. S. **Engenharia de software**: uma abordagem profissional. Porto Alegre: Amgh, 2016.

Para conhecer a resolução comentada proposta pelo professor, acesse a videoaula deste *Teoria em Prática* no ambiente de aprendizagem.



LEITURA FUNDAMENTAL

Indicações de leitura

Indicação 1

Entenda a estimativa em tempo de desenvolvimento do sistema e o cronograma nominal, conforme ajustes previstos pelo Cocomo II. Assim poderá entender por que é necessário ter a cautela em aumentar a quantidade de pessoas na equipe ágil.

Para realizar a leitura, acesse a plataforma Biblioteca Virtual da Kroton e busque pelo título da obra no parceiro Biblioteca Virtual 3.0_Pearson.

SOMMERVILLE, Ian. **Engenharia de Software**. São Paulo: Pearson Education do Brasil, 2018. Seção 23.6.5, p. 658-659.

Indicação 2

Leia a seção 35.4.1 do livro *Engenharia de software* e compreenda como uma tabela de riscos pode auxiliar na tomada de decisão das ocorrências, em função do impacto.

Para realizar a leitura, acesse a plataforma Biblioteca Virtual da Kroton e busque pelo título da obra no parceiro Minha Biblioteca.

PRESSMAN, R. S. **Engenharia de software**: uma abordagem profissional. Porto Alegre: Amgh, 2016. p. 783-785.

QUIZ

Prezado aluno, as questões do Quiz têm como propósito a verificação de leitura dos itens *Direto ao Ponto, Para Saber Mais, Teoria em Prática e Leitura Fundamental*, presentes neste *Aprendizagem em Foco*.

Para as avaliações virtuais e presenciais, as questões serão elaboradas a partir de todos os itens do *Aprendizagem em Foco* e dos slides usados para a gravação das videoaulas, além de questões de interpretação com embasamento no cabeçalho da questão.

1. De acordo com Sommerville (2018), se tem um projeto com previsão para 15 meses quando a equipe conta com quatro pessoas, significa que são 60 pessoas/mês, portanto, se incluir mais uma pessoa na equipe, é certeza que o mesmo sistema será entregue em 12 meses?

- a. Sim, pois o esforço de 60 pessoas/mês será dividido por cinco pessoas.
- b. Sim, com o aumento de uma pessoa, poderá ser entregue até antes de 12 meses.
- c. Não, pois a produtividade da equipe é prejudicada com o aumento da quantidade de pessoas na equipe.
- d. Sim, ao ingressar mais uma pessoa que já tem experiência e conhecimento do negócio, vai aumentar a velocidade da equipe.
- e. Não, pois não existe possibilidade de aumentar a quantidade de pessoas na equipe.

2. Na gestão de riscos, o gerente deve conhecer a probabilidade de ocorrência e, caso aconteça, qual será o impacto. Assinale a alternativa que preenche corretamente a afirmativa: "O gerente de um projeto de software deve dedicar menor esforço quando um fator de risco tem____, mas que a probabilidade de ocorrência é ____".


- a. Alto impacto / muito baixa
- b. Alto impacto; alta.
- c. Alto impacto; altíssima.
- d. Prejuízo; alta.
- e. Médio impacto; alta.



GABARITO

Questão 1 - Resposta C

Resolução: Sobre a duração de projeto e alocação de equipe, fica claro que o aumento do número de pessoas na



equipe pode diminuir a velocidade produtiva por pessoa pela necessidade de comunicação.

Questão 2 - Resposta A

Resolução: Segundo Pressman (2016, p. 785), quando a probabilidade é muito baixa, o gerente não deve dedicar grande esforços.

TEMA 3

Gestão da qualidade

Autoria: Marco Ikuro Hisatomi

Leitura crítica: Valéria Cristina Gomes Leal

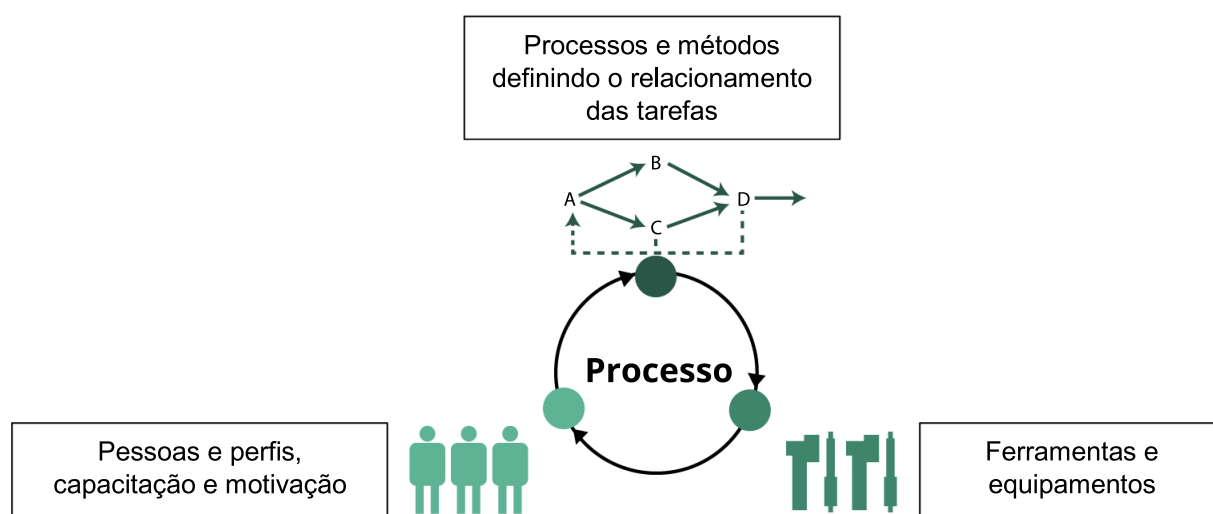


► DIRETO AO PONTO

Grande parte das implantações da gestão da qualidade nas organizações chegam ao fracasso pela falta de conhecimento das vantagens e do envolvimento nas decisões (PRESSMAN, 2016), assim causando a falta de apoio gerencial, orçamento mal dimensionado e, ainda, falta de aderência da própria equipe técnica por resistência cultural. Diante de todos os motivos e consequências, muitos dos responsáveis pela qualidade tentam, de alguma maneira, justificar o ganho que teria com o processo da qualidade e perfazem uma documentação extensa, tornando exageradamente formal, burocrático e mais complexa a ampliação do processo com as diretrizes da qualidade em software.

Para existir um equilíbrio na administração da organização, o que tornará possível a adoção da gestão da qualidade (CMMI DEVELOPMENT, 2010) é o esforço em conjunto de três dimensões críticas das organizações nas quais se concentram: pessoas, procedimentos e métodos, e ferramentas e equipamentos, conforme ilustrado na Figura 1.

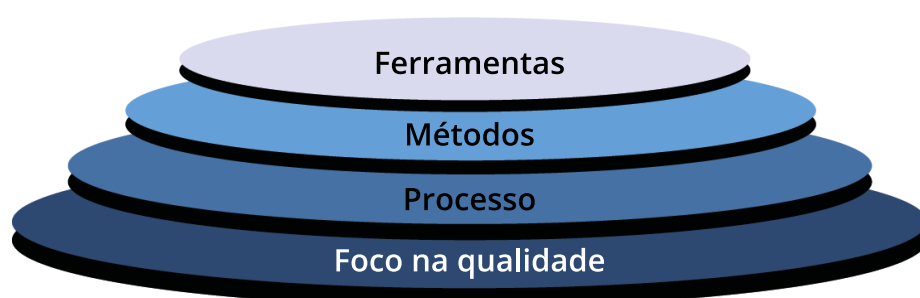
Figura 1 – Três dimensões críticas das organizações



Fonte: CMMI Development (2010, p. 4).

Portanto, a disciplina da gestão da qualidade está se tornando mais valorizada no processo de desenvolvimento de sistemas, acrescentando maior garantia de produtos seguros e úteis aos usuários finais. Normalmente, a qualidade é conquistada por meio do acúmulo de esforços, com o foco na qualidade (PRESSMAN, 2016), sendo este a base fundamental de sustentação da engenharia de software, conforme ilustra a Figura 2, e almejar a qualidade do produto de software impulsiona a implantação de um processo adequado aos recursos e conhecimentos da organização. Ainda, Pressman (2016) diz que podem-se utilizar métodos favoráveis aos projetos de software e as respectivas ferramentas que proporcionarão produtividade e qualidade no desempenho de habilidades técnicas e gerenciais.

Figura 2 – Camadas da engenharia de software



Fonte: Pressman (2016, p. 16).

Concluindo, aproveite a oportunidade para agregar maior conhecimento em gestão da qualidade, pois o profissional com esse perfil e que conhece um processo de software completo e métodos ágeis está valorizado.

Lembrando que o avanço rápido das tecnologias e a crescente complexidade das aplicações estão exigindo interfaces diversas (com muitos aplicativos, inclusive com equipamentos IoT¹), assim demandando empenho com o foco em qualidade durante o desenvolvimento de software.

¹ IoT é uma sigla de Internet of Things (Internet das Coisas), que possui a característica de conexão dos objetos cotidianos (carros, eletrodomésticos, etc.) para a comunicação de dados.

Referências bibliográficas

SOFTWARE ENGINEERING INSTITUTE. **CMMI® for Development**. Versão 1.3. Hanscom AFB, MA. 2010.
Disponível em: https://resources.sei.cmu.edu/asset_files/TechnicalReport/2010_005_001_15287.pdf. Acesso em: 7 jun. 2020.

PRESSMAN, R. S. **Engenharia de Software: uma abordagem profissional**. Porto Alegre: Amgh, 2016.

PARA SABER MAIS

Dentre as partes organizacionais, independente do modelo de processo, as pessoas são responsáveis pelo movimento rumo ao melhoramento contínuo do produto e do processo, por meio da percepção, comunicação, preparação, coordenação e do gerenciamento. O People-CMM² visa o aperfeiçoamento contínuo das pessoas por meio de práticas de trabalho nas organizações.

Embora o foco em qualidade de software está mais perceptível pelo esforço dedicado em engenharia de software e, especificamente, na gestão da qualidade, o SPI aponta como passo-chave a capacitação pessoal. Sendo assim, é importante que busque constantemente o desenvolvimento da habilidade pessoal, dando ênfase, em primeiro momento, às competências individuais, conforme a Figura 3, que ilustra uma escala de maturidade conforme *framework* People-CMM. Inicialmente, tenha a autodisciplina do aprendizado, conquistando o patamar de **treinamento e coordenação**, conseguindo realizar o gerenciamento do próprio desempenho. Avançando, poderá alcançar níveis organizacionais, em que as competências são

² O Modelo de Maturidade de Capacitação de Pessoas (People-CMM, People Capability Maturity Model) desenvolvido pela Software Engineering Institute (SEI).

verificadas pela inovação contínua da capacidade organizacional, com o envolvimento de todos, exercendo alinhamento do **desempenho organizacional**.

Figura 3 – Áreas de processo do People-CMM



Fonte: adaptado de Pressman (2016 p. 833).

Porém é imprescindível que, para alcançar o topo em habilidades, também possua as competências intermediárias demonstradas em **cultura participativa**, nível em que compartilha os conhecimentos e aprende com os demais da força de trabalho, assim, desenvolvendo o engajamento em equipe. Enquanto que, na escala da **capacidade organizacional**, encontrar-se-á contribuindo no gerenciamento do desempenho quantitativo organizacional e na integração das competências com os grupos de trabalho fortalecidos, principalmente em atividades da engenharia de software e no processo de desenvolvimento de software.

Referências bibliográficas

CMMI INSTITUTE. **People Capability Maturity Model**. Software Engineering Institute (SEI). Disponível em: <https://cmmiinstitute.com/getattachment/96aac30e-714b-4a32-8e8c-a91c54238474/attachment.aspx>. Acesso em: 7 jun. 2020.

PRESSMAN, R. S. **Engenharia de Software**: uma abordagem profissional. Porto Alegre: Amgh, 2016.

TEORIA EM PRÁTICA

A sua equipe está pronta para o desenvolvimento! Aliás, já desenvolveu vários softwares com razoável sucesso e agora está se preparando para adotar um modelo de qualidade. Ah sim, por onde e como começar? Sabe-se que um modelo pode ser implantado para dirigir sem modificar o que já sabem, apenas criando uma estrutura que conduza para a melhoria contínua. Bem, para Pressman (2016), a parte mais difícil é estabelecer um consenso para iniciar um processo de melhoria e definir uma estratégia contínua para implantá-la em uma empresa de desenvolvimento de software. Como você resolveria esse **desafio**? Quais são os princípios ou filosofia que deve adotar com a sua equipe? É necessário estar capacitado para esta implantação? Qual é o roteiro do SPI?

Para conhecer a resolução comentada proposta pelo professor, acesse a videoaula deste *Teoria em Prática* no ambiente de aprendizagem.

LEITURA FUNDAMENTAL

Indicações de leitura

Indicação 1

Uma leitura aprofundada em cenários diversificados tende a abrir os nossos horizontes para preparação do plano de qualidade.

Afinal, tudo que estamos desenvolvendo hoje será para o futuro, e este futuro é desconhecido, mesmo que outras pessoas já estejam praticando. Aquilo que não conhecemos ainda pode se tornar presente a qualquer momento, então recomendo um estudo sobre essa diversidade, conforme os itens: de código aberto, talentos da equipe, gestão da complexidade e demais tendências (algumas já em plena operação).

Para realizar a leitura, acesse a plataforma Biblioteca Virtual da Kroton e busque pelo título da obra no parceiro Minha Biblioteca.


PRESSMAN, R. S. **Engenharia de software:** uma abordagem profissional. Porto Alegre: Amgh, 2016. p. 843-849.

Indicação 2

O fundamento da qualidade e a sua garantia é a prática de um grupo pensando nas atividades da qualidade para planejar, coordenar, analisar e comunicar os resultados, segundo Pressman (2016), descritas nos tópicos 21.4 “Tarefas, metas e métricas da SQA”, juntamente com os tópicos relacionados, inclusive as abordagens formais.

Para realizar a leitura, acesse a plataforma Biblioteca Virtual da Kroton e busque pelo título da obra no parceiro Minha Biblioteca.

PRESSMAN, R. S. **Engenharia de software:** uma abordagem profissional. Porto Alegre: Amgh, 2016. p. 452-456.



QUIZ

Prezado aluno, as questões do Quiz têm como propósito a verificação de leitura dos itens *Direto ao Ponto, Para Saber Mais, Teoria em Prática e Leitura Fundamental*, presentes neste *Aprendizagem em Foco*.

Para as avaliações virtuais e presenciais, as questões serão elaboradas a partir de todos os itens do *Aprendizagem em Foco* e dos slides usados para a gravação das videoaulas, além de questões de interpretação com embasamento no cabeçalho da questão.

1. Baseado no *framework* SPI, quais são os passos fundamentais para implantação da gestão da qualidade e de processo de melhoria contínua?
 - a. Reconhecer o estado atual, qualificação, escolha e instalação do modelo, mensurar resultados.
 - b. Reconhecer o estado atual, escolha e instalação do modelo, mensurar resultados.
 - c. Treinamento e capacitação, reconhecer o estado atual, escolha e instalação do modelo, mensurar resultados.
 - d. Reconhecer o estado atual, qualificação, escolha e instalação do modelo.
 - e. Reconhecer o estado atual, mensurar resultados, escolha e instalação do modelo, capacitação pessoal.
2. Podem ocorrer falhas na implantação de um processo de qualidade em processo de software, por conta de vários fatores, levando ao fracasso um projeto baseado

no SPI para a melhoria contínua do desenvolvimento de software, tal como: ____.

Escolha a alternativa que preenche corretamente a lacuna.

- a. Aderência total da equipe técnica.
- b. Motivação cultural.
- c. Falta de apoio gerencial.
- d. Documentação adequada do processo.
- e. Apoio da alta gestão.



GABARITO

Questão 1 - Resposta A

Resolução: O SPI se torna eficiente ao proceder um ciclo de atividades para impulsionar a organização na instalação do processo de melhoria contínua (PRESSMAN, 2016, p. 823). O conteúdo da filosofia foi ministrado no Bloco 4 dos slides.

Questão 2 - Resposta C

Resolução: São motivos de fracasso: a falta de apoio gerencial, orçamento mal dimensionado, falta de aderência da própria equipe técnica, por resistência cultural, uma documentação extensa tornando exageradamente formal.

TEMA 4

Administração da manutenção do software

Autoria: Marco Ikuro Hisatomi

Leitura crítica: Valéria Cristina Gomes Leal

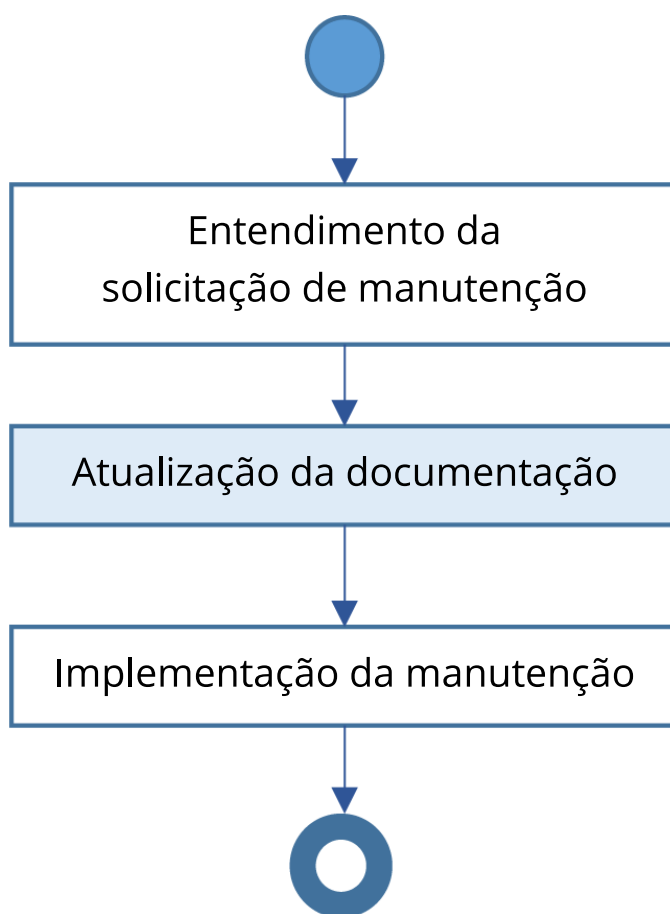


► DIRETO AO PONTO

Alguns processos da fase de manutenção do software podem receber atenção especial para aprimoramento nos procedimentos e longevidade ao sistema, com qualidade e custos menores da manutenção.

Dentre eles podemos citar a atualização da documentação sistematizada, conforme ilustra a Figura 1, e adotar o processo iterativo, principalmente por meio do método XP¹.

Figura 1 – Ênfase em documentação



Fonte: elaborada pelo autor.

¹ XP (programação extrema) pode ser consultado em Pressman (2016, p. 72), item 5.4 “*Extreme programming*”.

Caso existam documentações dos sistemas, devem permanecer consistentes durante toda vida útil (compreendendo as duas fases: desenvolvimento e manutenção), ou devem ser reavaliadas para adotar uma metodologia que possa, de alguma maneira, **assegurar que os artefatos de documentação estejam atualizados.**

Segundo Taentzer *et al.* (2019, p. 11), são muitos documentos em um processo de manutenção do software: especificações de requisitos, modelos de análises, design (dos arquitetos de software), códigos escritos em paralelo (em diretórios e arquivos auxiliares), casos e roteiros de testes, entre outros documentos. Lembrando ainda que os artefatos de documentação dos sistemas possuem assuntos inter-relacionados, por isso a necessidade de uma validação consistente entre todos eles. Exemplo: seria bem custoso manter atualizado um texto com uma linguagem de modelagem ou natural e semiestruturada, um dos artefatos comuns de serem usados em atividade de programação.

Outra ação importante está relacionada ao procedimento de implementação/programação por meio da adoção de técnicas dos métodos ágeis. Numa fase de evolução do sistema, Sommerville (2018, p. 236) destaca que “o novo código deve ser refatorado e melhorado para evitar a degradação do programa”, assim aumentará o nível de manutenibilidade, que é um dos fatores da qualidade num sistema que esteja sofrendo atualizações e sendo útil para a organização; de acordo com Taentzer *et al.* (2019), entende-se que, ao refatorar, favorecerá consideravelmente as tarefas evolução e correção na fase de manutenção do software.

Referências bibliográficas

PRESSMAN, R. S. **Engenharia de software**: uma abordagem profissional. Porto Alegre: Amgh, 2016.

SOMMERVILLE, I. **Engenharia de software**. São Paulo: Pearson Education do Brasil, 2018.

TAENTZER, G. *et al.* (Ed.) **Managed software evolution** – The nature of software evolution. Cham, Switzerland: Springer Open, 2019. Disponível em: <https://doi.org/10.1007/978-3-030-13499-0>. Acesso em: 11 jun. 2020.

PARA SABER MAIS

Vamos entender por que os testes de regressão são apropriados na fase de manutenção do software. Especialmente os testes de regressão, tanto pelo histórico de dados coletados durante a fase de operação em que o sistema se encontra quanto pelo foco na entrega da versão modificada num ambiente operacional em plena produção.

Na administração da manutenção do software, os testes de regressão proporcionarão produtividade e eficácia, aumentando a qualidade da entrega para maior confiança aos usuários. Dentre os fatores para avaliação da aplicação, Sommerville (2018) destaca os dados de teste recorrendo ao histórico de testes de regressão após a implementação de mudanças no sistema, seja por adição, alteração ou eliminação de funcionalidades.

Sommerville (2018, p. 245) ressalta também quanto ao aumento da base de dados, em quantidade de registros e alterações/ inclusão de estruturas de dados, fatos que podem aumentar as

inconsistências dos dados e, ainda, muitas vezes, é improvável a eliminação dos dados inconsistentes, apesar de serem antigos.

Compreenda o objetivo e uso do teste de regressão em Pressman (2016, p. 478), para uma estratégia de redução de “efeitos colaterais” quando é implementada alterações ou inclusão de novos componentes no software. Porém, Pressman (2016) ressalta o cuidado com o aumento do volume de dados de testes, sobretudo deve cobrir os casos de testes de acordo com as regras de negócio, portanto, Pressman (2016, p. 479) direciona:

[...] a criação de conjuntos de testes de acordo com classes de casos de testes: a) Uma amostra representativa dos testes que usam todas as funções do software; b) Testes adicionais que focalizam as funções de software que podem ser afetadas pela alteração e c) Testes que focalizam os componentes do software que foram alterados.

Embora a gestão da qualidade seja um processo largamente útil na fase de desenvolvimento de software, vale muito se beneficiar deste recurso também na fase de manutenção: pela facilidade em obter dados reais para os casos de teste e pela iminência de se instalar uma versão modificada diretamente no ambiente operacional em produção, visto que o impacto, por falhas, será bem maior do que na fase de transição na primeira implantação do sistema.

Referências bibliográficas

PRESSMAN, R. S. **Engenharia de software**: uma abordagem profissional. Porto Alegre: Amgh, 2016.

SOMMERVILLE, I. **Engenharia de software**. São Paulo: Pearson Education do Brasil, 2018..



TEORIA EM PRÁTICA

Em uma situação, a organização está com o sistema em operação, ou seja, o software foi entregue com sucesso, sendo que existiu a validação das funcionalidades de acordo com as especificações da fase de análise e design. Todos os documentos, instruções de operação e sistemas foram devidamente instalados nos respectivos servidores da organização, tendo a garantia que a versão em uso é estável. Muito bem, agora vamos entender como administrar a demanda de solicitação de manutenção.

Se a equipe de manutenção e evolução do software é a mesma, será menos custosa e mais rápida a implementação da solicitação de melhoria. Mas e se, por acaso, **a equipe que efetuou o desenvolvimento e a entrega é diferente da equipe responsável pela evolução?**

Sendo assim, qual atividade é importante e fundamental antes que o sistema sofra a implementação da manutenção?

Ao decidir pela implementação da manutenção, o que deve acontecer com a documentação do sistema?

Para conhecer a resolução comentada proposta pelo professor, acesse a videoaula deste *Teoria em Prática* no ambiente de aprendizagem.

LEITURA FUNDAMENTAL

Indicações de leitura

Indicação 1

A **gestão de configuração** é um conjunto de atividades que permite rastrear e controlar cada item/artefato pertencente a uma *baseline* (linha base) numa determinada versão em que o sistema foi disponibilizado. Faça a leitura desse recurso considerado importante para a administração da manutenção do software, que permite juntar a qualquer momento todos os itens/artefatos de uma determinada versão para uma verificação e validação. A leitura está disponível no livro de *Engenharia de software*, itens: 29.1 “Gestão de configuração de software” e 29.2 “O repositório de SCM”.

Para realizar a leitura, acesse a plataforma Biblioteca Virtual da Kroton e busque pelo título da obra no parceiro Minha Biblioteca.


PRESSMAN, R. S. **Engenharia de software**: uma abordagem profissional. Porto Alegre: Amgh, 2016. p. 624-632.

Indicação 2

Um assunto relacionado à manutenção do software que deve ter conhecimento é a reengenharia, em especial a **reengenharia de software** quanto aos principais tipos e o objetivo de cada uma delas, conforme descrito por Pressman (2016), no item 36.5 “Reengenharia de software”.

Para realizar a leitura, acesse a plataforma Biblioteca Virtual da Kroton e busque pelo título da obra no parceiro Minha Biblioteca.

PRESSMAN, R. S. **Engenharia de software**: uma abordagem profissional. Porto Alegre: Amgh, 2016. p. 802-805.



QUIZ

Prezado aluno, as questões do Quiz têm como propósito a verificação de leitura dos itens *Direto ao Ponto, Para Saber Mais, Teoria em Prática e Leitura Fundamental*, presentes neste *Aprendizagem em Foco*.

Para as avaliações virtuais e presenciais, as questões serão elaboradas a partir de todos os itens do *Aprendizagem em Foco* e dos slides usados para a gravação das videoaulas, além de questões de interpretação com embasamento no cabeçalho da questão.

1. Pressman (2016, p. 804) afirma que “o tipo mais comum de reengenharia é _____. Alguns sistemas legados têm uma arquitetura de programa razoavelmente sólida, mas os módulos individuais foram codificados de uma maneira que dificulta entendê-los, testá-los e _____. Nesses casos, _____ dentro dos módulos suspeitos pode ser reestruturado”. Escolha a alternativa que preenche corretamente as lacunas.
 - a. A reestruturação de código; mantê-los; o código.
 - b. A reengenharia reversa; mantê-los; a arquitetura.
 - c. A reestruturação dos documentos; atualizá-los; os comentários.
 - d. A reestruturação dos dados; alterá-los; as estruturas.
 - e. A engenharia direta; reestruturá-los; o código.

2. Sommerville (2018, p. 235) diz que “quando há envolvimento de times diferentes, uma diferença fundamental entre o desenvolvimento e a evolução é que

o primeiro estágio da implementação da mudança requer uma compreensão do programa”.

Qual alternativa corresponde melhor ao objetivo dessa compreensão do programa citado pelo autor?

- a. Analisar o custo e o esforço necessário por parte dos usuários para testar o sistema.
- b. Analisar o impacto da mudança nas documentações de operação do sistema para o usuário final.
- c. Analisar o impacto da mudança para certificar-se de que não afetará outra parte do sistema ou a integração com outros sistemas do mesmo ambiente operacional.
- d. Criar um cenário de teste para ser aplicado antes da entrega.
- e. Analisar os sistemas de mobile, principalmente porque existem muitas instalações.



GABARITO

Questão 1 - Resposta A

Resolução: A alternativa A é a única que condiz com o texto do autor citado e também representa o conceito correto sobre o assunto.

Questão 2 - Resposta C

Resolução: Segundo o autor, tecnicamente, o engenheiro de software deve conhecer o suficiente para manter os níveis de qualidade quanto ao funcionamento do sistema e das integrações com outros sistemas.



BONS ESTUDOS!