DEDMD

LIVRO

UNIDADE 3

Engenharia de Software

Gerenciamento de Qualidade de *Software*

Roque Maitino Neto

2016
Editora e Distribuidora Educacional S. A.
Avenida Paris, 675 – Parque Residencial João Piza
CEP: 86041 -100 — Londrina — PR
e-mail: editora.educacional@kroton.com.br
Homepage: http://www.kroton.com.br/

Sumário

GERENCIAMENTO DE QUALIDADE DE *SOFTWARE*

Convite ao estudo

Olá! Seja bem-vindo à terceira unidade da disciplina de Engenharia de Software. Nas duas primeiras unidades deste material você teve a oportunidade de conhecer dois modos distintos, quase antagônicos, de desenvolvimento de *software*: o modelo tradicional e o modelo ágil. Antes deles, no entanto, você foi conduzido até os fundamentos da Engenharia de *Software* e à teoria associada ao processo de desenvolvimento de um produto; bases necessárias para sua entrada no universo das metodologias de desenvolvimento que seriam abordadas na sequência.

Conforme o conteúdo teórico era apresentado, a história da X-Soft era contada. No início, tínhamos uma empresa pequena e sem metodologia formal implantada. Com a chegada de um grande cliente, surgiu também a necessidade de organização dos seus processos e da adoção de um modelo de desenvolvimento para seu produto. A metodologia Cascata foi introduzida e, passado algum tempo, o *Extreme Programming* passou a ser o modelo oficial de desenvolvimento.

É fato que a X-Soft evoluiu. No entanto, o modelo utilizado, por melhor e mais moderno que seja, não dispensa uma providência elementar: o gerenciamento da qualidade do produto. O objetivo geral desta unidade é que você se torne capaz de gerenciar por completo a qualidade do produto e do processo utilizado para criá-lo. Cada seção, em específico, tem os objetivos que seguem:

- Abordar os fundamentos de qualidade e da gestão da qualidade do *software*;
- Abordar os fatores que afetam a qualidade e as ações de Garantia da Qualidade do *Software* (SQA);
- Abordar as normas de qualidade mais referenciadas;
- Introduzir métricas e inspeções de *software*.

Como consequência do atingimento desses objetivos, você deverá desenvolver competência técnica para conhecer e conduzir processos de qualidade de *software*. Você atingirá os objetivos e desenvolverá as competências desta unidade por meio da resolução da realidade profissional em quatro etapas, descritas aqui:

1. Criar relatório contendo o levantamento das práticas de qualidade atualmente implementadas;

2. Criar relatório contendo o levantamento dos fatores que afetam a qualidade dos produtos;

3. Criar relatório contendo o processo de implantação de norma de qualidade na X-Soft;

4. Criar relatório contendo o processo de implantação de inspeções, medições e métricas de *software*.

Nas próximas páginas serão detalhados a primeira parte de nossa missão, o conteúdo teórico proposto sobre qualidade de *software* e novas abordagens da situação-problema.

Bom trabalho!

Seção 3.1

A gestão da qualidade no processo de desenvolvimento de software

Diálogo aberto

O conteúdo teórico apresentado na unidade 2 abordou três das principais metodologias ágeis usadas para desenvolvimento de *software* e ajudou você a trilhar o caminho até a efetiva implantação do *Extreme Programming* na X-Soft.

O modo ágil de se conduzir um projeto já foi incorporado ao cotidiano da equipe e as entregas seguem sendo feitas, na grande maioria das vezes, no prazo e no limite do orçamento combinados. O encantamento do cliente, de certa forma a grande meta a ser alcançada pela direção da empresa, tem sido verificado por meio de bons retornos dados à equipe por quem a contratou.

O aprimoramento da qualidade do processo e do produto deve ser, no entanto, desafio constante na vida da X-Soft e de qualquer outra empresa. Por esse motivo – e por alguns outros que lhe serão apresentados no decorrer desta unidade – ações específicas de garantia da qualidade do produto devem ser tomadas. Afinal, a obtenção de produto de qualidade é uma das justificativas da existência da própria Engenharia de *Software* como disciplina estruturada.

Para que seu aproveitamento nesta seção e nas seguintes seja ótimo, há questões importantes a serem respondidas: afinal, o que é qualidade? Estamos tratando de uma percepção subjetiva ou de um conceito puramente objetivo? Um bom produto criado há 30 anos seria hoje considerado um produto de qualidade? Intrigante, não acha?

Utilizando os temas que estão sendo abordados nesta seção e sua crescente habilidade em diagnosticar situações e procedimentos ativos na empresa, você deve criar um relatório contendo o levantamento das práticas de qualidade atualmente implementadas. Nesse relatório devem ser descritas as ações atualmente tomadas para que o produto da X-Soft esteja adequado ao seu propósito e em conformidade com os requisitos.

Na seção "Não pode faltar" você terá contato com fundamentos de qualidade de *software* e sua gestão. Um excelente aproveitamento dos seus estudos é o que desejamos a você.

Não pode faltar

Não se pode conceber um produto, qualquer que seja sua natureza, que não seja criado levando-se em conta sua qualidade. A busca por níveis satisfatórios e previamente definidos de excelência deve basear qualquer processo de fabricação em larga escala ou de manufatura, sob pena de se obter produto com baixa aceitação de mercado.

Nesta seção, você terá contato com o conceito de qualidade e com sua gestão no processo de desenvolvimento de um *software*. Entender a importância da qualidade é o primeiro passo em direção à sua elevação de condição indispensável para o sucesso de um produto. Sigamos adiante!

Conceito de qualidade

O que é qualidade de *software* e por que ela é tão importante? Uma boa maneira de começarmos esta discussão é afastando a ideia de que qualidade signifique perfeição. É comum entendermos que sempre será possível encontrar algo a ser melhorado em algo que se reconhece como de boa qualidade.

Também não se pode caracterizar a qualidade como algo absoluto, definitivo e que tenha meios de ser medida universalmente, em parâmetros aceitáveis para todas as pessoas. O que parece ser de boa qualidade para mim pode não parecer para você. E vice-versa.

Por ser considerada um conceito de muitas dimensões, talvez uma boa maneira de começarmos a entender a qualidade, é que esta se realiza por meio de um conjunto de características e atributos de um certo produto. Como qualidade não significa perfeição, é natural que tenhamos que estabelecer um nível aceitável de qualidade para um produto e meios para verificar se esse nível foi alcançado.

Embora saibamos que traços de subjetividade e a perspectiva própria do avaliador farão parte de uma avaliação, o "nível aceitável" de qualidade precisa ser o mais objetivo possível, extraído por meio de processos estruturados e ferramentas apropriadas de medição de qualidade.

Ao longo dos anos, autores e organizações têm definido o termo qualidade de formas diferentes. Para Crosby (1979, p. 23), qualidade é a "conformidade com os requisitos do usuário". Para ele, se o produto atende aos requisitos explícitos e implícitos, significa que o produto é de qualidade. Por exemplo, ao comprarmos uma

televisão, se os nossos requisitos estiverem de acordo com as suas características, o aparelho então nos serve. Caso contrário, escolhemos alguma outra que esteja mais próxima de nossas expectativas.

Humphrey (1989, p. 280) refere-se à qualidade como "a conquista de excelentes níveis de adequação ao propósito", enquanto a IBM cunhou a frase "qualidade dirigida ao mercado", que se baseia na conquista total da satisfação do cliente.

A expressão "adequação ao propósito" pressupõe a existência de um registro da descrição do propósito do produto. Tomando-se como exemplo um secador de cabelos, seu próprio nome carrega sua funcionalidade. Características técnicas são colocadas num manual de instruções. No caso de um *software*, seu propósito está inserido na especificação de requisitos.

Mais recentemente, a qualidade foi definida pela norma ISO9001-00 como "o grau no qual um conjunto de características inerentes preenche os requisitos".

Assimile

Observe outros dois conceitos e ideias relacionados à qualidade:

"Qualidade de Software é um processo sistemático que focaliza todas as etapas e artefatos produzidos com o objetivo de garantir a conformidade de processos e produtos, prevenindo e eliminando defeitos" (BARTIÉ, 2002, p. 16).

"Qualidade é a totalidade das características de um produto de software que lhe confere a capacidade de satisfazer necessidades implícitas e explícitas" (ISO/IEC 9126-1, 2003, p. 17).

Você já deve ter se perguntado: o que define, afinal, um bom *software*? É de se esperar que nesta resposta apareçam os elementos que compõem o conceito de qualidade dos quais tratamos há pouco. Uma das possíveis medidas de qualidade é, de fato, a adequação ao propósito, o que significa que o *software* funciona efetivamente de acordo com o que foi projetado. Shaffer (2013) propõe uma forma interessante de se medir a qualidade de um *software* que é o seu valor de mercado, especialmente em situações em que alguém está procurando investir na empresa que o produz. É muito comum também que as pessoas percebam a qualidade de um *software* como reflexo da qualidade do processo usado para criá-lo.

Embora você possa não encontrar uma definição universal e definitiva para a qualidade aplicada a um *software*, fatores como a corretude, a eficiência e a usabilidade são algumas medidas amplamente aceitas como indicadores da qualidade do produto. Esses fatores serão estudados com mais propriedade e detalhes na próxima seção, mas vale uma olhada agora:

• **Corretude**: capacidade do *software* em executar suas funcionalidades conforme elas foram definidas. Se pudéssemos resumir esse fator em uma pergunta, ela seria próxima de: "o software faz aquilo que eu quero?"

• **Eficiência**: relaciona-se ao grau de adequação do programa aos recursos de *hardware*, tais como processador e memória. Eficiência é uma palavra muito comumente usada, embora muitas vezes de forma incorreta. Para ele, eficiência é a medida de quantos recursos são usados para que uma tarefa seja completada. Atualmente, com o *hardware* custando tão pouco, não se presta muita atenção no fator eficiência como no passado, exceto em processos que incluem quantidade muito grande de dados, como o *Big Data*, por exemplo (SHAFFER, 2013).

• **Usabilidade**: este fator está relacionado com a facilidade de uso do produto. Em outras palavras, trata-se da medida da capacidade do público-alvo em obter valor do *software* por meio da sua interface.

• **Portabilidade**: é possível usá-lo em outra plataforma? Trata-se da medida de facilidade em mudar o *software* de uma plataforma (*Windows*, por exemplo) para outra (*Mac*, por exemplo).

• **Interoperabilidade**: trata-se da "capacidade de diversos sistemas e organizações trabalharem em conjunto (interoperar) de modo a garantir que pessoas, organizações e sistemas computacionais interajam para trocar informações de maneira eficaz e eficiente". (Disponível em: <http://www.governoeletronico.gov.br/acoes-e-projetos/e-ping-padroes-de-interoperabilidade/o-que-e-interoperabilidade>. Acesso em: 18 fev. 2016).

Com essas características, você começa a entender como a qualidade de um produto é avaliada. Nas páginas seguintes você terá contato com aspectos da gestão da qualidade do software, colocada em duas diferentes perspectivas. Sigamos adiante.

Pesquise mais

Você encontrará questões interessantes sobre qualidade no artigo publicado na "II Escola Regional de Informática da Sociedade Brasileira

de Computação Regional de São Paulo – II ERI da SBC" – Piracicaba, SP – junho de 1997, p. 173-189. Disponível em: <https://xa.yimg.com/kq/groups/21646421/371309618/name/Modelos+de+Qualidade+de+Software.pdf>. Acesso em: 18 jan. 2016.

Gestão da qualidade do software

Com a finalidade de conseguirmos um melhor aproveitamento neste tema, ficaria melhor se definirmos a gestão da qualidade como um sistema. Idealmente esse sistema de gestão da qualidade já deve estar incorporado na organização e, como se espera de qualquer sistema, ele deve incluir processos, pessoas e ferramentas dirigidas à obtenção da qualidade nas entregas.

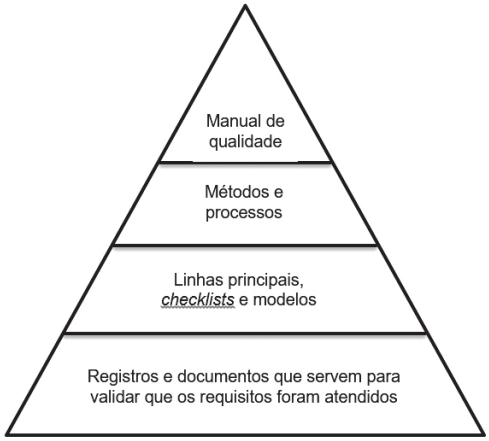
Quando tomamos um sistema de gerenciamento financeiro ou um sistema de gerenciamento de pessoas em uma organização como exemplos, por si sós eles não deveriam demandar uma nova equipe de gerenciamento ou novos recursos. Ao contrário disso, espera-se que tais sistemas sejam partes integrantes da organização, alinhadas com necessidades particulares de finanças ou de gestão de pessoas.

De acordo com Moorthy (2013), um sistema de gestão de qualidade de *software* deve possuir 4 níveis: o nível 1 é composto pelo manual de qualidade da empresa; o nível 2 refere-se aos métodos e processos usados pela equipe para entregar suas tarefas; o nível 3 contém as linhas principais, os *checklists* e os modelos, usados com bastante frequência no dia a dia e importantes na manutenção da consistência das informações e, por fim, o nível 4 refere-se aos registros e documentos usados para fins de validação de um produto, usados como evidências de uma atividade e úteis para referência futura. A figura 3.1 mostra a organização dos níveis de um sistema de gestão de qualidade.

De acordo com SWEBOK (2004), publicação criada pela IEEE (Instituto de Engenheiros Eletricistas e Eletrônicos, do inglês *Institute of Electrical and Electronics Engineers*), o gerenciamento da qualidade de *software* é tratado como um processo, que se aplica a todas as perspectivas de processos de *software*, produtos e recursos. Ele define os requisitos e os responsáveis pelos processos, suas medições e saídas, além dos canais de *feedback*. O planejamento da qualidade do *software* envolve:

- A definição do produto em termos de suas características de qualidade;
- O planejamento do processo para se obter o produto desejado.

Figura 3.1 – Níveis de um sistema de gestão de qualidade de *software*



Fonte: MOORTHY. **Jumpstart to Software Quality Assurance**. 2013, p.184.

Pois bem, um processo específico de gestão de *software* é definido no padrão IEEE 12207.0-96 e inclui o **Processo de garantia da qualidade**, mais conhecido pela abreviatura SQA (*Software Quality Assurance* ou Garantia da Qualidade do *Software*).

Esse processo – também composto por várias etapas – visa assegurar que os produtos de *software* e os processos que os constroem estão em conformidade com os requisitos (ou funcionalidades) por meio de planejamento e execução de um conjunto de atividades destinadas a transmitir a certeza de que a qualidade é fator integrante do produto. Isso significa que a equipe poderá se assegurar de que o problema foi clara e suficientemente compreendido e que os requisitos da solução foram definidos e expressos de forma adequada. Em relação ao processo, o papel do SQA é assegurar que tudo será implementado de acordo com o plano traçado. Desafiador, não acha?

Reflita

Sabemos que um processo de gerenciamento de qualidade tem por objetivo a tomada de providências que incluam a qualidade em todas as ações executadas durante o ciclo de desenvolvimento de um produto. No entanto, como garantir que esse processo foi concebido e aplicado corretamente? Como medir a qualidade desse processo? Em que momento a equipe poderá entender que construiu, de fato, algo de qualidade?

Outro processo que visa conferir qualidade ao produto é o que chamamos de **Verificação e Validação**. Para facilitar as referências ao tema, os termos verificação e validação são tratados como apenas um. De acordo com SWEBOK (2004), trata-se de um processo bem estruturado para avaliar os produtos de *software* em todo o seu ciclo de vida, do planejamento até sua efetiva entrega. Em resumo, retrata o esforço da equipe para garantir que a qualidade está embutida no *software* e que ele reflete o desejo do usuário. A V&V, como também é conhecido esse processo, está interessada diretamente na qualidade do produto.

O grupo **revisões e auditorias**, nosso último processo de qualidade abordado aqui e também tratado como um único item, inclui dois principais procedimentos:

• **Revisões técnicas**: o objetivo de uma revisão (ou análise) técnica é o de avaliar um produto de *software* para determinar a sua adequação para a sua utilização pretendida. O objetivo é o de identificar discrepâncias a partir das especificações e dos padrões aprovados. Os resultados devem fornecer evidências que confirmem (ou não) que o produto atende às especificações;

• **Inspeções**: o propósito de uma inspeção é detectar e identificar anomalias no *software*. Esta prática se diferencia das revisões em dois aspectos: alguém que exerça cargo de gestão sobre qualquer membro da equipe de inspeção não deverá participar desse processo, e uma inspeção deve ser conduzida por um facilitador imparcial, treinado em técnicas de inspeção.

As inspeções incluem também um líder, um responsável pelos registros da seção e um número reduzido de inspetores, comumente de 2 a 5 pessoas.

Na próxima seção, dedicada exclusivamente ao estudo da SQA e dos fatores que afetam a qualidade do *software*, você conhecerá de forma mais profunda e abrangente as providências que uma equipe pode tomar para garantir a qualidade do seu produto de *software* e a satisfação do cliente. Na sequência, seu desafio prático para esta seção será melhor definido e uma solução possível lhe será indicada.

Exemplificando

Um bom exemplo de como um setor pode se organizar para obter software de qualidade, é dado pelo caso em que determinado segmento do setor agropecuário, junto com profissionais de TI, estabeleceu um conjunto de características que foram destacadas como imprescindíveis para avaliação da qualidade de um software agropecuário. Em linhas gerais, as características eleitas foram: facilidade de uso (a interface é facilmente

personalizada), facilidade de operação (é simples a entrada de dados de natureza física, zootécnica, financeira no *software*?) e integridade do sistema (o programa é capaz de manter o processamento, a despeito da ocorrência de ações inesperadas?), entre outras (ROCHA; MALDONADO; WEBER, 2001).

Faça você mesmo

No decorrer desta seção foi feita menção à Verificação e Validação como uma das providências para assegurar a qualidade do *software*. Será que V&V são atividades redundantes ou complementares? Executadas em conjunto, podem ser consideradas como teste? Em outras palavras, Verificação + Validação = Teste? Pesquise mais sobre o tema e sintetize o que aprendeu a respeito em relatório.

SEM MEDO DE ERRAR

As práticas ágeis implantadas na X-Soft têm assumido importância crescente no processo de desenvolvimento da empresa e se mostrado eficientes no aprimoramento da relação com os clientes. Sabemos que as entregas vêm sendo feitas de forma satisfatória, mas não há na X-Soft ainda processo definido e formalizado de gerenciamento da qualidade do produto. Isso significa que, embora os programas atendam à maioria dos requisitos estabelecidos, as providências de qualidade tomadas pela equipe durante sua construção baseiam-se muito mais na percepção subjetiva de seus membros do que em práticas e medidas objetivas e consagradas pela prática.

Neste cenário, fica muito difícil que a equipe e seus gerentes consigam estabelecer níveis aceitáveis de qualidade do produto, já que não contam com meios formais para medi-la.

No item "Diálogo Aberto" foi proposto como desafio para esta seção o levantamento das práticas de qualidade atualmente implementadas na X-Soft e a geração de relatório contendo tal levantamento.

Uma solução possível para esse desafio é a que segue:

1. O relatório deve ser iniciado com a descrição de seu objetivo, que é o de levantar as práticas de qualidade vigentes na empresa;

2. Um relato de elementos-chave da equipe deve ser colhido. Nesse relato deverá estar presente o que cada um pensa da qualidade e como a coloca em prática;

3. Na sequência, o relatório deverá descrever como atualmente são feitos os testes no produto, com que frequência, em quais ocasiões e por quem;

4. Por fim, o relatório deverá explicitar os pontos falhos nas investidas da equipe em favor da qualidade, tais como falta de revisões nos artefatos liberados, a frequência insuficiente na aplicação de testes e a falta de documentação adequada, por exemplo.

Atenção!

A implantação de um processo formal de qualidade pode, num primeiro momento, transmitir a impressão de que tempo precioso de desenvolvimento está sendo desperdiçado em medições, auditorias, inspeções e práticas afins. Cabe aos líderes da equipe conscientizá-la da importância desse processo.

Lembre-se

Embora a maioria dos envolvidos com Tecnologia da Informação sejam capazes de entender a importância da qualidade, é sempre bom reforçar seu valor no processo. Leia o artigo encontrado em: <http://www.devmedia.com.br/qualidade-de-software-parte-01/9408>. Acesso em: 11 jan. 2016, e reforce sua crença na qualidade.

Avançando na prática

Pratique mais	
Instrução Desafiamos você a praticar o que aprendeu transferindo seus conhecimentos para novas situações que pode encontrar no ambiente de trabalho. Realize as atividades e depois as compare com as de seus colegas.	
Aprimorando as práticas atuais para aprimorar a qualidade	
1. Competência Geral	Conhecer as principais metodologias de desenvolvimento de *software*, normas de qualidade e processos de teste de *software*.
2. Objetivos de aprendizagem	O objetivo desta seção é apresentar os fundamentos da qualidade e da gestão da qualidade do *software*.
3. Conteúdos relacionados	A gestão da qualidade no processo de desenvolvimento de *software*.

4. Descrição da SP	Uma empresa desenvolvedora de soluções de *software* tem estado parcialmente satisfeita com a qualidade dos seus produtos. Embora as entregas tenham sido na maioria das vezes satisfatórias, faz parte de seus objetivos de curto prazo a redução de ocorrência de falhas e o consequente aprimoramento dos programas. Há, no entanto, certa relutância da direção em adotar procedimento formal de qualidade, já que seus membros entendem que tal providência ocasionaria mudanças bruscas na rotina da equipe e investimento de esforço em processos sem retorno garantido. Seu desafio é indicar, no âmbito do modelo ágil, ações para melhorar o nível de qualidade dos produtos sem que grandes mudanças na equipe e em seus procedimentos sejam feitas.
5. Resolução da SP	A solução do caso requer as possíveis providências: • Durante o planejamento do programa, na fase em que o cliente descreve as funcionalidades desejadas para o programa, deverá haver sempre mais do que um responsável por validar tais funcionalidades, dispensando atenção especial para a viabilidade de sua implantação; • Uma vez validadas as funcionalidades desejadas, também deverá haver rígido acompanhamento do tempo e dos recursos alocados para o cumprimento delas. É comum que estimativas incorretas levem ao apressamento do desenvolvimento e a consequente queda no nível de qualidade do produto; • Acompanhamento rigoroso do processo de testes por parte do analista de testes. O cliente deverá ser orientado a criar testes adequados a cada funcionalidade. Por outro lado, a equipe deverá testar continuamente cada funcionalidade antes de sua liberação.

Lembre-se

"A qualidade é um aspecto que deve ser tratado simultaneamente ao processo de desenvolvimento do software, pois ela não pode ser imposta depois que o produto está finalizado" (ROCHA; MALDONADO; WEBER, 2001, p. 67).

Faça você mesmo

Qual o perfil de um profissional de qualidade? Qual sua atuação? Faça a leitura do artigo publicado em: <http://www.tiegestao.com.br/2013/09/01/carreira-analista-de-teste-o-que-e-e-o-que-faz/>. Acesso em: 20 jan. 2016.

Faça valer a pena!

1. No contexto de um processo de gestão da qualidade de um software, assinale a alternativa que contém expressões que completam corretamente as lacunas na frase abaixo.

"De acordo com a IEEE, o gerenciamento da qualidade de software é tratado como _____, que se aplica a todas as perspectivas de processos de software, produtos e recursos. _____ da qualidade do software envolve a definição do produto em termos de suas características de _____ e o planejamento do processo para se obter o produto desejado".

a) uma opção, O desenvolvedor, confiabilidade

b) um processo empírico, A perspectiva, requisitos

c) um processo, O responsável, qualidade

d) um processo, O planejamento, qualidade

e) uma opção, A persecução, requisitos

2. Em relação aos aspectos gerais do conceito de qualidade, analise as afirmações que seguem:

I) Pode-se considerar que um dos aspectos da qualidade é a conformidade do produto com os requisitos para ele estabelecidos.

II) O conceito de qualidade é universal e absoluto e em relação a ele não existem divergências.

III) Como o nível de excelência que se deseja para um programa é alto, qualidade deve ser sinônimo de perfeição.

IV) No âmbito da Tecnologia da Informação, a qualidade deve ser considerada como um conceito puramente subjetivo.

É verdadeiro o que se afirma apenas em:

a) I e IV

b) II e III

c) I

d) II

e) II e IV

3. Em relação ao Processo de Garantia da Qualidade do Software (SQA), analise as afirmações que seguem:

I) Visa assegurar que os produtos construídos estão totalmente livres de erros.

II) Visa assegurar que os produtos de software e seus processos estão em conformidade com os requisitos.

III) Em relação ao processo, o SQA visa assegurar que tudo será implementado de acordo com o plano traçado.

É verdadeiro o que se afirma apenas em:

a) III

b) II

c) I e III

d) II e III

e) I

Seção 3.2

A garantia da qualidade do software

Diálogo aberto

Seja bem-vindo de volta!

Em nossa última seção, tivemos a oportunidade de fazer o primeiro contato com um assunto que nos acompanhará até o final do curso: a qualidade relacionada ao produto de *software*, ou seja, a um programa.

Mesmo depois da introdução ao tema, muitas questões ainda ficaram para serem respondidas. Por ora, as que mais nos interessam são as seguintes: o que torna o programa um bom programa? Quais são os parâmetros de qualidade? O entendimento da importância desse tema é bem fácil de ser alcançado. Basta você considerar que, se a dependência das organizações tem aumentado em relação ao uso de *software*, é de se esperar que a exigência sobre a qualidade dos produtos tenha aumentado na mesma proporção.

A X-Soft, claro, não deixará de dispensar atenção adequada a essa demanda. Seu corpo gerencial sabe que a implantação de estratégia de qualidade não é apenas uma opção a ser cogitada, mas uma ação necessária à sua própria sobrevivência.

Visando a adoção de procedimento formal de qualidade, você foi chamado a ser o responsável por dar início ao processo de mudança, levantando e relatando o que de efetivo tem sido feito em prol da excelência dos programas criados pela empresa.

Com esse diagnóstico em mãos, o próximo passo será realizar a apuração do nível dos indicadores de qualidade dos programas. Para que esse desafio seja superado com a excelência de sempre, você terá à disposição na seção "Não pode faltar" o conteúdo sobre os fatores que afetam a qualidade de um produto e os procedimentos de garantia da qualidade de um *software*.

Por meio deles, você poderá criar um relatório contendo o nível em que se encontra cada um dos fatores que afetam a qualidade dos produtos da X-Soft. Por meio dessa prática, você desenvolverá competência para identificar, uma a uma, as características a serem aprimoradas em um programa, meio pelo qual será possível torná-lo integralmente adequado ao seu propósito, e dessa forma, poderá conhecer e conduzir processos de qualidade de *software*. Eis o desafio.

Bom trabalho!

Não pode faltar

Na seção anterior a esta, foram discutidos conceitos e temas introdutórios da qualidade, indispensáveis para a formação da sua percepção sobre o tema. O caminho que levará você à compreensão dos mecanismos envolvidos num processo de qualidade passa agora pela Garantia da Qualidade do *Software* (SQA) e pelo estudo mais aprofundado dos fatores que exercem influência sobre a qualidade de um produto de *software*. Nos próximos parágrafos, esses assuntos serão abordados de maneira objetiva e dirigida ao melhor aproveitamento do tema para fins de torná-lo apto a transformar o desafio e as questões propostas em missões cumpridas. Siga adiante!

Garantia da Qualidade de Software (SQA)

Uma definição aceitável para a SQA (*Software Quality Assurance*) é expressa como "padrão planejado e sistemático de ações que são exigidas para garantir a qualidade do *software*" (PRESSMAN, 1995, p. 733). Em outras palavras, são as providências tomadas pela equipe para assegurar a qualidade do seu produto, de maneira vinculada ao processo que o cria.

Como a abrangência temporal de tais providências se estende das fases iniciais do processo até a finalização do *software*, é esperado que a responsabilidade das equipes do projeto e demais envolvidos também seja extensa. Por exemplo, os engenheiros de *software*, o profissional que gerencia o projeto, o grupo de SQA e o próprio cliente, todos eles são responsáveis pela garantia da qualidade.

A definição de SQA inclui a expressão "padrão planejado e sistemático de ações", não é mesmo? Pois bem, vamos então às ações.

Atividades SQA

Assegurar a qualidade de um *software* envolve algumas tarefas, três das quais são descritas na sequência (PRESSMAN, 1995):

• **Aplicação de métodos técnicos**: a SQA começa a ser aplicada desde a especificação e o desenho do sistema. Uma especificação malfeita – ou uma história mal escrita pelo cliente ou mal interpretada pela equipe – certamente irão comprometer a qualidade do produto final. Assim que uma especificação, protótipo ou *release* de um sistema tiverem sido criados, será apropriado avaliá-los quanto à qualidade.

• **Realização de revisões técnicas formais**: esta é a atividade central no contexto da avaliação da qualidade de um produto. Uma revisão técnica formal é um encontro no qual uma equipe (de 3 a 5 pessoas, normalmente) destacada para o trabalho, concentra-se na busca por problemas de qualidade no produto ou, mais comumente, numa parte específica dele. Elas são aplicadas em momentos variados do processo e são também usualmente referenciadas como *walkthrough* (passo a passo). Se você é fã de *games*, então já ouviu ou leu essa expressão.

• **Atividades de teste de *software***: por melhor que seja sua técnica, por maior que seja sua capacidade de fazer bons programas e por mais que você siga uma metodologia, submeter seu programa a um processo de teste nunca sairá de moda. O motivo é simples: errar é humano.

O teste é uma atividade desempenhada para avaliar a qualidade do produto e para sua melhoria, por meio da identificação de defeitos e problemas. O teste de *software* consiste na verificação dinâmica do comportamento de um programa em um conjunto finito de casos de teste, adequadamente selecionado a partir de um número geralmente infinito de execuções do programa (SWEBOK, 2004). Parece complicado? Como a próxima unidade será inteira dedicada ao assunto, trataremos aqui apenas de alguns elementos essenciais do teste.

O processo de teste envolve quatro etapas: planejamento, projeto de casos de teste, execução e avaliação dos resultados, que devem ser conduzidas ao longo de todo o processo de desenvolvimento (ROCHA; MALDONADO; WEBER, 2001). Testar, não se trata, portanto, de vasculhar o código linha por linha procurando falhas no programa. Tampouco é executar o programa algumas vezes procurando por erros. Curioso para saber mais? Retomaremos o assunto na seção 3.4, que dará destaque para teste de *software*.

Fatores que influenciam na qualidade do *software*

Já que temos tratado de processos formais de qualidade na construção do nosso conhecimento sobre o tema, nada mais apropriado do que a busca por um modelo de qualidade consagrado para definirmos as características desejáveis em um programa. O modelo de qualidade ISO/IEC 25010:2011 estabelece um conjunto de oito características internas e externas de um *software*, divididas em outras tantas subcaracterísticas. Para facilitar a compreensão global da norma e facilitar sua síntese, a tabela 3.1 apresenta tais características de qualidade, separadas entre próprias do produto e próprias do uso, conforme será explicado na sequência.

Tabela 3.1 – Modelo de qualidade da ISO 25010:2011

Tipo	Características	Subcaracterísticas
Características do produto	Adequação funcional	Completude funcional, Corretude funcional (acurácia) e funcionalidade apropriada.
	Confiabilidade	Maturidade, disponibilidade, tolerância a falhas e recuperabilidade.
	Usabilidade	Apropriação reconhecível, inteligibilidade, operabilidade, proteção contra erro do usuário, estética de interface com o usuário e acessibilidade.
	Eficiência de Desempenho	Comportamento em relação ao tempo, utilização de recursos, capacidade.
	Segurança	Confidencialidade, integridade, não repúdio, rastreabilidade de uso e autenticidade.
	Compatibilidade	Coexistência e Interoperabilidade.
	Capacidade de manutenção	Modularidade, reusabilidade, analisabilidade, modificabilidade, testabilidade.
	Portabilidade	Adaptabilidade, instalabilidade e substituibilidade.
Características de uso	Efetividade	Efetividade
	Eficiência	Eficiência
	Satisfação	Utilidade, prazer, conforto e confiança.
	Uso sem riscos	Mitigação de risco econômico, mitigação de risco a saúde e segurança e mitigação de risco ambiental.v
	Cobertura de contexto	Completude de contexto e flexibilidade.

Fonte: Wazlawick (2013, p. 233).

Os parâmetros de qualidade normalmente são segmentados entre os que possuem mais afinidade com o processo, com o produto percebido pelo usuário ou com o produto sob o ponto de vista da equipe. As medidas de qualidade internas, por exemplo, servem para avaliar aspectos que usualmente são percebidos apenas pelos desenvolvedores. A capacidade de manutenção e a facilidade em se aplicar testes são bons exemplos dessas medidas.

As medidas de qualidade externas alcançam características avaliadas pela equipe do ponto de vista do usuário. Por exemplo, a eficiência e capacidade de operação de um programa.

O modelo ISO/IEC 25010:2011, cujas características foram resumidas na tabela acima, agregou as características internas e externas num único grupo e o chamou de **características do produto**. Elas podem ser avaliadas no ambiente de desenvolvimento, ao passo que as características do ***software* em uso** podem apenas ser avaliadas durante o efetivo uso do sistema.

Parece bastante desafiador construir um produto que tenha boa adequação a todas essas características, não acha?

Pesquise mais

A ISO (Organização Internacional para Padronização ou Organização Internacional de Normalização) é uma organização independente e não governamental, presente em 162 países. Por meio de seus membros, ela desenvolve padrões internacionais aplicáveis em diversas áreas da atividade humana. Pesquise mais sobre a ISO em seu site oficial <http://www.iso.org/iso/home/about.htm>. Acesso em: 14 jan. 2016. Site em inglês ou no Portal Educação: <http://www.portaleducacao.com.br/administracao/artigos/40732/a-historia-da-organizacao-iso#!1>. Acesso em: 28 jan. 2016.

Para que você possa adquirir preparo para superar o desafio proposto, vale conhecer melhor algumas das características de qualidade mencionadas na tabela 3.1.

1. Adequação funcional: antes conhecida como funcionalidade apenas; ela se refere à existência de um conjunto de funções que satisfazem às necessidades previamente estabelecidas, quando o produto é usado sob condições específicas. Duas de suas subcaracterísticas são a Completude Funcional (o *software* apresenta todas as funções necessárias ao usuário?) e a Corretude Funcional ou Acurácia (o *software* gera dados e consultas corretos segundo o que foi definido?) (WAZLAWICK, 2013).

2. Confiabilidade: se um *software* é capaz de manter comportamento consistente com o que se espera dele ao longo do tempo, então ele pode ser considerado confiável. A confiabilidade tem a ver com o funcionamento do programa em situações incomuns. Ela pode ser medida diretamente e estimada usando-se dados históricos e de desenvolvimento, ou seja, um computador poderá ser considerado livre de falhas quando não tiver alguma incidência em um determinado ambiente e num determinado período (MUSA et al., 1987 apud PRESSMAN, 1995). Desta definição, o que ainda não temos claro é o que significa *falha*. Quando tratarmos de teste de *software* com mais detalhes, esse conceito será abordado.

Vale aqui também destacar duas de suas subcaracterísticas: a Disponibilidade (avalia o quanto o *software* está operacional e livre para uso quando necessário) e a Tolerância a Falhas (avalia a forma como o *software* reage em situação anormal).

Reflita

Qual é o real interesse de um usuário? Quando ele pensa em qualidade do software, em geral, ele se lembra da confiabilidade. No caso de o produto já ser relativamente confiável, a usabilidade é que fará diferença nele.

3. Usabilidade: de forma simplificada, podemos entender essa característica como a facilidade em se usar um programa, do ponto de vista do usuário. Em linhas gerais, o programa é fácil de usar se ele é (REISS, 2012):

• Funcional – ele realmente funciona?

• Responsivo – ele me fornece respostas adequadas?

• Ergonômico – eu posso facilmente ver, clicar, arrastar e girar as coisas?

• Conveniente – tudo está bem onde eu preciso que esteja?

• "À prova de tolos" – o projetista me ajuda a não cometer erros ou quebrar coisas?

A usabilidade também apresenta subcaracterísticas, incluindo (WAZLAWICK, 2005):

• Operabilidade – o produto é fácil de usar e controlar?

• Proteção contra erro do usuário – o programa consegue evitar que o usuário cometa erros?

• Acessibilidade – avalia o grau em que o produto foi projetado para atender usuários com necessidades especiais.

Assimile

"Não é possível conceber um processo de garantia de qualidade de um *software* sem integrá-lo com o ciclo de desenvolvimento de software" (BARTIÉ, 2002, p. 35).

4. Segurança: se um programa consegue proteger os dados e as funções de acessos não autorizados, então ele tem um bom nível de segurança. Algumas de suas subcaracterísticas devem ser destacadas. Entre elas:

• Confidencialidade – mede o grau em que os dados e funções ficam disponíveis para quem, de fato, tem autorização para acessá-los;

• Rastreabilidade de uso – mede o grau em que as ações realizadas por uma pessoa ou por um sistema podem ser rastreadas de forma a ser efetivamente comprovado que, de fato, foi essa pessoa ou sistema que realizou tais ações.

5. Capacidade de manutenção: trata de uma característica que atrai interesse direto apenas da equipe de desenvolvimento, já que não afeta a percepção do usuário

em relação ao sistema. Como você certamente já inferiu, trata-se da capacidade do sistema em passar por manutenção. Assim como todas as outras características apresentadas, essa também conta com divisões. Vamos a elas:

• Modularidade – o sistema é bem dividido em módulos? Mudanças em um dos módulos deve causar mínimo impacto nos outros.

• Reusabilidade – há partes do sistema que podem ser usadas na construção de outro sistema?

• Analisabilidade – o sistema permite que se faça depuração com facilidade?

Exemplificando

Ainda em dúvida sobre a importância da qualidade em um produto? Em 4 de junho de 1996, o Foguete Ariane 5, construído pela empreiteira EADS SPACE *Transportation*, explodiu 37 segundos após seu lançamento. A comissão responsável pela apuração dos fatos concluiu que houve erro no Sistema Referencial Inercial (SRI). Do relatório, consta o seguinte: "A anomalia interna de software do SRI ocorreu durante a execução de uma conversão de dados de um número de 64 bits em ponto flutuante para um inteiro de16 bits com sinal. O valor do número em ponto flutuante era maior do que poderia ser representado pelo inteiro de 16 bits com sinal. O resultado foi um operando inválido. A instrução de conversão de dados não estava protegida contra erros de operando".

Disponível em: <http://www.ime.uerj.br/~demoura/Especializ/Ariane/>. Acesso em: 17 jan. 2015.

No âmbito das características de qualidade do *software* em uso, vale destacar (WAZLAWICK, 2013):

6. Efetividade: capacidade que o produto tem de proporcionar ao cliente o atingimento de seus objetivos de negócio.

7. Satisfação: capacidade que o produto tem de satisfazer o cliente em ambiente de uso. Pode ser dividida em Utilidade, Prazer, Conforto e Confiança.

8. Uso sem riscos: o uso do *software* não pode implicar riscos para pessoas, negócios ou ambiente. Divide-se em mitigação do risco econômico, mitigação do risco ambiental e mitigação do risco à saúde e segurança.

Faça você mesmo

O método *Cleanroom* (sala limpa) é uma maneira bastante difundida de se desenvolver *softwares* com qualidade. O método foi publicado pela primeira vez em 1987 por Mills, Dyer e Linger, e tem como princípio básico que os programas devem ser vistos como regras para funções ou relações matemáticas. A atividade aqui proposta é a criação de um relatório de uma página contendo os princípios básicos do modelo. Vale a pena conhecê-lo.

Na sequência, nosso desafio será retomado e uma solução viável para ele será discutida.

SEM MEDO DE ERRAR

Os ventos da mudança passaram novamente pela X-Soft carregando o desejo – e a real necessidade – de implantação de procedimento formal de qualidade na empresa. A primeira providência tomada foi o levantamento do que já tem sido feito em prol da qualidade. Convenhamos, nada melhor que iniciar uma mudança entendendo como as coisas funcionam atualmente.

No entanto, apenas essa providência não basta. Mesmo que em pequenos passos, a implantação da qualidade demanda ações contínuas, tanto processuais como culturais. Ela requer a conscientização de que sempre haverá por onde melhorar um produto ou um processo e que a excelência é um bem que, embora intangível, produz efeitos palpáveis e duradouros no crescimento e na boa imagem da empresa.

Pois bem, o desafio agora é tratar da avaliação dos fatores que afetam a qualidade dos produtos da X-Soft. Com base no conteúdo estudado nesta seção, você deve produzir relatório contendo a sua avaliação de algumas características de um *software* produzido pela X-Soft. Como esse *software* não chegou a ser efetivamente construído, devemos considerar que as avaliações devam ser apenas estimadas, de acordo com os critérios que definem cada uma delas. Vejamos:

1. O relatório deve ser iniciado com a descrição de seu objetivo, que é o de avaliar as principais características de qualidade de um programa produzido pela empresa.

2. Na sequência, você deverá descrever a avaliação das seguintes características do sistema, buscando responder perguntas relacionadas a cada uma delas:

2.1. Capacidade de manutenção: o programa favorece a manutenção? O sistema possui boa divisão de módulos? É fácil de ser testado?

2.2. Segurança: o programa permite acesso apenas de pessoas autorizadas? Ele oferece condições de rastrear seu uso?

2.3. Usabilidade: o programa é fácil de ser usado? Sua operação é de fácil assimilação?

2.4. Adequação funcional: o *software* apresenta todas as funções necessárias ao usuário? Ele gera dados e consultas corretos segundo o que foi definido?

Com esse levantamento, você terá um bom panorama da qualidade do produto e poderá direcionar esforços de melhoria com mais precisão.

Atenção!

A característica conhecida como "Uso sem riscos" não deve ser confundida com a característica "Segurança do *software*".

Lembre-se

Alguns dos objetivos das Revisões Técnicas Formais são (i) descobrir erros de implementação em qualquer representação do *software*; (ii) verificar se o *software* atende seus requisitos e (iii) garantir que o *software* tenha sido representado de acordo com os padrões definidos (PRESSMAN, 1995).

Avançando na prática

Pratique mais	
Instrução Desafiamos você a praticar o que aprendeu transferindo seus conhecimentos para novas situações que pode encontrar no ambiente de trabalho. Realize as atividades e depois as compare com as de seus colegas.	
Melhorando a capacidade de manutenção dos programas	
1. Competência Geral	Competência para conhecer e conduzir processos de qualidade de *software*.
2. Objetivos de aprendizagem	O objetivo desta seção é abordar os fatores que afetam a qualidade e as ações de Garantia da Qualidade do *software* (SQA).
3. Conteúdos relacionados	Garantia da qualidade de *software*. Fatores que afetam a qualidade de um *software*.

4. Descrição da SP	A M-Soft, empresa desenvolvedora de soluções de *software* para consultório médico, atingiu bons níveis de qualidade em seus programas e, com isso, tem obtido bons níveis de satisfação de seus clientes. Em seus produtos, as características que afetam diretamente a experiência e as expectativas do usuário têm sido muito bem avaliadas. Contudo, qualquer necessidade de alteração ou aprimoramento nos produtos tem causado preocupação e trabalho excessivo à equipe, pois aparentemente eles não têm sido bem preparados para sofrer manutenção. O desafio aqui é indicar ações que aumentem a capacidade de manutenção dos próximos sistemas produzidos pela M-Soft, tornando-os mais flexíveis e aptos para sofrerem mudanças. Consulte a Seção 1.3 da Unidade 1 para relembrar os conceitos de Coesão e Acoplamento e separação de dados e procedimentos, importantes na resolução deste desafio.
5. Resolução da SP	A solução do caso demanda as possíveis providências: 1. Aprimoramento da modularidade do *software*. A divisão lógica dos componentes que executam as funções deve ser feita de forma a se obter alta coesão e baixo acoplamento entre os módulos; 2. O projeto deverá conter representação distinta de dados e procedimentos; 3. Criação ou aprimoramento das interfaces que reduzam a complexidade das conexões entre módulos e o ambiente externo; 4. A reutilização de componentes já testados em outras aplicações deve ser fortemente considerada.

Lembre-se

"O teste de *software* é uma das atividades de validação e verificação e consiste na análise dinâmica do mesmo, ou seja, na execução do produto de *software* com o objetivo de verificar a presença de defeitos no produto e aumentar a confiança de que o produto esteja correto" (ROCHA; MALDONADO; WEBER, 2001, p. 73).

Faça você mesmo

É possível avaliar a qualidade de um *software* se o cliente ficar mudando de ideia sobre o que espera que o *software* faça? Sintetize sua opinião e discuta-a com seus colegas de sala.

Faça valer a pena!

1. Em relação aos fatores que influenciam a qualidade de um software, analise as afirmações que seguem:

I) Pode-se considerar que um dos fatores que diretamente afetam o produto é a prática exercida pela equipe de se reunir mais do que uma vez por dia.

II) A característica de Segurança de um sistema está ligada ao seu uso seguro.

III) A Satisfação é um item de qualidade que se mede pelo tempo que o usuário permanece operando o sistema.

IV) Os parâmetros de qualidade são divididos entre os que possuem mais afinidade com o processo e com o produto.

É verdadeiro o que se afirma apenas em:

a) I e IV

b) II e III

c) II

d) IV

e) II e IV

2. Assinale a alternativa que contém apenas expressões relacionadas aos fatores que influenciam na qualidade do software.

a) Quantidade de linhas de código, compactação do sistema.

b) Conjunto de funções que satisfazem a necessidades previamente estabelecidas, probabilidade de operação livre de falhas de um programa.

c) Disponibilidade de funções substitutas, facilidade de manutenção.

d) Existência de procedimento formal de comercialização do sistema, contratação de profissional de programação com experiência comprovada.

e) Alteração do comportamento estável durante o uso, quantidade de linhas de código.

3. No contexto da usabilidade de um software, assinale a alternativa que contém expressões que completam corretamente as lacunas na frase abaixo.

"A usabilidade avalia o grau no qual o produto tem atributos por meio dos quais possa ser _____, _____, usado e que seja atraente ao usuário. Em específico, a _____ avalia o grau no qual o produto é fácil de usar e controlar. A _____ deve proporcionar prazer e uma interação satisfatória."

a) projetado, testado, acessibilidade, rapidez

b) entendido, aprovado, inteligibilidade, praticidade

c) visualizado, rastreado, operabilidade, experiência

d) simulado, prototipado, informalidade, coloração

e) entendido, aprendido, operabilidade, interface com o usuário

Seção 3.3

As normas de qualidade aplicadas no desenvolvimento de software

Diálogo aberto

Olá! Seja bem-vindo a mais esta seção.

Já discutimos na seção anterior que a percepção sobre o que se costuma chamar de qualidade de um programa pode – e de fato irá – variar de usuário para usuário. No entanto, como você pode imaginar, apenas a subjetividade não é suficiente para a construção de base sólida para a realização de avaliações precisas de um programa. Para que se possa distinguir um bom produto de outro que ainda pode ser melhorado, é necessário o estabelecimento de normas gerais e objetivas de qualidade, universalmente aceitas e capazes de permitir comparações confiáveis entre características correspondentes dos programas.

Depois de levantar as ações que eram em dado momento executadas em prol da qualidade e de avaliar algumas características de seus programas, a X-Soft agora foca seus esforços na adoção de uma norma de qualidade que seja capaz de estabelecer parâmetros seguros de avaliação e de proporcionar à empresa o necessário reconhecimento como uma desenvolvedora com altos níveis de excelência.

O desafio que lhe é proposto nesta seção é o de criar um relatório contendo o processo de escolha e implantação de norma de qualidade na X-Soft. Para que você possa contar com a base teórica necessária para superá-lo, as próximas páginas trarão os fundamentos das principais e mais aceitas normas de qualidade de *software*, o que dará suporte para a escolha daquela a ser implantada na organização.

Por meio do conhecimento adquirido com essa prática, você terá condições para selecionar o modelo (ou norma) de qualidade que mais se adequa à realidade da sua organização, tornando-a tão bem-sucedida quanto possível na arte de desenvolver programas de computador.

Que o caminho para a escolha do modelo comece a ser trilhado.

Boa sorte e bom trabalho!

Não pode faltar

Em quais parâmetros podemos confiar na hora de escolhermos um modelo de qualidade? Qual instituição cria os padrões, os divulga e os mantém? Há uma norma melhor que a outra? Esta seção do seu livro didático aborda essas questões e alguns dos mais bem-conceituados modelos de qualidade, com destaque para seus pontos fortes e aplicações. Antes de nos debruçarmos sobre eles, vale a pena sabermos um pouco mais sobre a organização que os cria e mantém.

Em Genebra, cidade da Suíça, está localizado o escritório central de uma organização independente e não governamental, com membros em 162 países e que desde fevereiro de 1947 já publicou mais de 20.500 padrões internacionais que cobrem quase todos os aspectos dos ramos da atividade humana, tais como saúde, agricultura, manufatura e, é claro, tecnologia. Essa organização é mundialmente conhecida como ISO, sigla da *International Organization for Standardization*. Por conta da existência de diferentes acrônimos (ou siglas) em línguas diferentes, tais como IOS em inglês e OIN em francês, os fundadores resolveram adotar simplesmente ISO, palavra derivada do grego *isos*, que significa igual Disponível em: <http://www.iso.org/iso/home/about.htm>. Acesso em: 23 jan. 2016.

Estudaremos na sequência alguns dos principais padrões de qualidade utilizados mundo afora. Eles darão a você base para superar o desafio proposto.

ISO 9001 – Sistema de Gestão da Qualidade

A ISO 9001 é um dos mais conhecidos e utilizados padrões mundiais de qualidade. Atualizado no ano de 2015, ele especifica requisitos para um sistema de gestão de qualidade, com foco naquilo que o cliente exige para que o produto ou serviço seja entregue de acordo com suas necessidades (SEEAR, 2015).

Esse padrão é implantado quando uma organização precisa demonstrar sua capacidade de fornecer produtos e serviços que atendam às exigências de regulamento e estatuto da organização e que pretende aumentar a satisfação do cliente por meio da aplicação eficaz do sistema. O aspecto interessante desse padrão é que todos os seus requisitos são genéricos e se destinam a aplicação em qualquer organização, independentemente da sua natureza ou tamanho, ou dos serviços ou produtos que disponibiliza. (Disponível em: <http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=62085>. Acesso em: 23 jan. 2016). Não acha que é disso que a X-Soft precisa?

Pesquise mais

A ISO 9001 faz parte da família de normas ISO 9000 e, como todas elas, tem passado por atualizações regulares. Rocha, Maldonado e Weber (2001), em sua obra *Qualidade de Software: Teoria e Prática*, oferecem bom panorama das atualizações recentes, exceto a do ano de 2015. Esta última, aliás, pode ser conhecida em: <http://www.qualityconsult.com.br/index.php/iso-9001-versao-2015/>. Acesso em: 23 jan. 2016.

A ISO 9001:2015 adota uma abordagem de processo para desenvolvimento, implementação e melhoria da eficácia de um sistema de gestão da qualidade, com o objetivo de aumentar a satisfação do cliente por meio do atendimento aos seus requisitos. Para isso, ela se baseia em 7 princípios de gerenciamento de qualidade: Foco no Cliente, Liderança, Engajamento de Pessoas, Abordagem de Processo, Melhoria, Tomada de Decisão Baseada em Evidências, Gestão de Relacionamento.

Esses termos, se considerados de forma isolada, podem não transmitir por completo o "espírito" do padrão. No entanto, pense no foco no cliente, na liderança e nas outras expressões num contexto de processo de qualidade no qual a satisfação dos requisitos colocados pelo cliente é prioridade, a participação efetiva das pessoas é fundamental e a melhoria – seja qual for o produto – é objetivo constante.

ISO/IEC 90003 – Orientações para Qualidade de Processo de Software

Nosso último item de estudo tratou de aspectos gerais da ISO 9001 e a colocou como um padrão geral para gestão da qualidade. A ISO/IEC 90003:2014 fornece orientações para aplicação da ISO 9001 nos processos de aquisição, fornecimento, desenvolvimento, operação e manutenção de um programa de computador e serviços de suporte relacionados.

A aplicação desse padrão é apropriada para produtos de *software* que são parte de um contrato comercial com uma outra organização, para produtos disponíveis para um segmento de mercado, produtos usados para dar suporte a um processo em uma empresa, produtos incorporados em um *hardware* específico ou produtos relacionados a um serviço de *software* Disponível em: <http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=66240>. Acesso em: 24 jan. 2016. Ou seja, trata-se de um padrão de qualidade aplicável a uma gama considerável de programas.

Reflita

A implantação de um padrão de qualidade em uma empresa pequena é necessária? Alguns argumentos talvez ajudem na resposta dessa questão: conquista de novos mercados, criação de condições para o surgimento de clima de incentivo, motivação e participação, direcionamento da empresa para o cliente e até redução de custos.

O padrão ISO/ IEC 90003:2014 não está atrelado à tecnologia, modelos de ciclo de vida, processos de desenvolvimento, sequência de atividades e estrutura organizacional.

Que tal um modelo que ofereça melhorias no gerenciamento do processo de *software* e conduza à produção de programas mais confiáveis? Sigamos adiante.

CMMI

O CMMI, sigla de *Capability Maturity Model Integration* é o sucessor do CMM, mantido pela SEI (*Software Engineering Institute*) de 1987 até 1997. Trata-se de um modelo que visa aprimorar a capacidade da maturidade do processo de *software*. Para melhor entendê-lo, voltemos à nossa primeira unidade. Na segunda seção, conceituamos assim processo: sequência de passos que visam a produção e manutenção de um *software* e que se inter-relacionam com recursos (humanos e materiais), com padrões, com entradas e saídas e com a própria estrutura da organização. Pois bem, a capacidade e maturidade de um processo remete à noção do grau de qualidade com o qual um processo atinge um resultado esperado Disponível em: <http://www.devmedia.com.br/cmmi-uma-visao-geral/25425#ixzz3yBwjcLs1>. Acesso em: 24 jan. 2016.

A versão atual do CMMI, a 1.3, possui três divisões:

• CMMI-ACQ: modelo que foca em atividades de gerenciamento da aquisição de produtos e serviços de *software*.

• CMMI-DEV: criado para o desenvolvimento de produtos e serviços de qualidade.

• CMMI-SVC: modelo que tem o objetivo de fornecer serviços de qualidade para clientes e usuários finais.

O CMMI é um caminho de melhoramento evolucionário, trilhado em 5 níveis de maturidade e 4 níveis de capacidade, feito para que organizações de software possam sair de um processo de *software* imaturo para um processo maduro e disciplinado. A tabela 3.2 mostra esses níveis:

Tabela 3.2 – níveis de capacidade e maturidade do CMMI

Nível	Capacidade	Maturidade
0	Incompleto	-
1	Realizado	Inicial
2	Gerenciado	Gerenciado
3	Definido	Definido
4	-	Quantitativamente gerenciado
5	-	Em otimização

Fonte: Wazlawick (2013, p. 506)

Segundo a descrição do modelo, a qualidade é influenciada pelo processo e seu foco é melhorar o processo de uma organização.

Existem duas estruturas do CMMI: em estágios e contínua. Embora a diferença seja sutil, ela é significativa. A representação (ou estruturação) em estágios usa níveis de maturidade para caracterizar o estado geral dos processos da organização em relação ao modelo como um todo, enquanto a representação contínua utiliza níveis de capacidade para caracterizar o estado dos processos da organização em relação a uma área de processo individual. Em outras palavras, a representação por estágios é aplicada à organização como um todo e permite que se compare a maturidade de diferentes organizações. Já a representação contínua é projetada para permitir à empresa focar em processos específicos que deseja melhorar em função de suas prioridades (WAZLAWICK, 2013).

Assimile

O CMM foi de fato criado pela SEI, que é o Instituto de Engenharia de *Software* da Carnegie Mellon University, mas contou com a participação do governo e da indústria norte-americana. Atualmente o CMMI tem sido mantido pelo CMMI Institute, que também passou a ser responsável pelo treinamento e certificação no modelo.

A essência do modelo é a divisão da capacidade e da maturidade em níveis, conforme mostrado na tabela 3.2. Vejamos os níveis em detalhes Disponível em: <http://cmmiinstitute.com/system/files/models/CMMI_for_Development_v1.3.pdf>. Acesso em: 24 jan. 2016.

Níveis de Capacidade do Processo: um nível de capacidade do processo é alcançado quando todos os objetivos genéricos para aquele nível são satisfeitos.

Nível 0 – Incompleto: um processo incompleto é um processo que não está

sendo colocado em prática ou que é usado apenas parcialmente. Um ou mais objetivos específicos da área de processo não estão sendo satisfeitos e inexistem metas genéricas para serem alcançadas, daí não haver razão para tornar oficial um processo parcialmente executado.

Nível 1 – Realizado: este nível é caracterizado como um processo que está sendo seguido, é capaz de gerar produtos, é viável no atingimento de metas específicas e já pode ser considerado como um fator de melhorias na organização. No entanto, tais melhorias podem ser perdidas ao longo do tempo, já que o processo ainda não foi institucionalizado.

Nível 2 – Gerenciado: um processo no nível 2 já é planejado e executado de acordo com a política definida, emprega pessoas qualificadas que tenham recursos adequados para produzir saídas controladas; envolve as partes interessadas (os *stakeholders*), é monitorado, controlado e avaliado. A disciplina do processo refletido por este nível ajuda a garantir que as práticas existentes são mantidas durante períodos de estresse.

Nível 3 – Definido: trata-se de um processo gerenciado que é feito sob medida a partir do conjunto de processos padrão e diretrizes da organização. Este nível também se caracteriza por manter registros de descrição do processo. No nível de capacidade de 2 (Gerenciado), as normas, as descrições de processos e os procedimentos podem ser bastante diferentes em cada instância específica do processo. No nível de capacidade 3, as normas, descrições de processos e procedimentos para um projeto são feitos sob medida a partir do conjunto de processos padrão da organização para atender um determinado projeto ou unidade organizacional e, portanto, são mais consistentes.

Outra distinção importante é que, no nível de capacidade 3, processos são geralmente descritos de forma mais rigorosa do que no nível 2. Um processo definido estabelece claramente a finalidade, os insumos, os critérios de entrada, as atividades, os papéis de cada membro, medidas, saídas e critérios de saída. Enfim, no nível 3, os processos são geridos de forma mais proativa e disciplinada.

Exemplificando

Uma boa indicação de como o CMMI tem sido bem aceito é dada por sua utilização em grandes organizações. No link: <http://cmmiinstitute.com/who-uses-cmmi>, acesso em: 24 jan. 2016, você poderá conferir que NASA, Siemens, Bosch, Motorola e IBM, entre outras empresas de ponta, adotaram o modelo em seus processos.

Em Língua Portuguesa, outro bom exemplo de como algumas práticas do CMMI podem ser aplicadas em pequenas e médias empresas de software, sem que o negócio se torne financeiramente inviável, pode ser encontrado em: <http://www.techoje.com.br/site/techoje/categoria/detalhe_artigo/1734>. Acesso em: 5 fev. 2016.

Outros bons exemplos dos benefícios trazidos pela adoção do CMMI são demonstrados em documento preparado pela SEI (*Software Engineering Institute* ou Instituto de Engenharia de Software) e disponibilizado em: <http://resources.sei.cmu.edu/asset_files/SpecialReport/2003_003_001_14117.pdf>. Acesso em: 5 fev. 2016. O relatório aborda 12 casos que apontam melhorias obtidas por empresas em suas medidas de desempenho em custo, cronograma, qualidade, satisfação do cliente e no retorno do investimento. Tomemos como exemplo as melhorias obtidas nas medidas de satisfação do cliente e aprimoramento da qualidade, expressas nos seguintes indicadores:

1. Aumento de 55% nas vitórias em concorrências públicas em comparação com CMMI nível 2 de maturidade por parte da Lockheed Martin;

2. Redução de defeitos encontrados de 6,6 para 2,1 em cada mil linhas de código, por parte da Northrop Grumman;

3. Mais de US$ 2 milhões de dólares economizados por causa da detecção e remoção prematura de defeitos no código, por parte da Sanchez Computer Associates, Inc.

Níveis de Maturidade: para dar suporte à representação por estágios, o CMMI contempla níveis de maturidade em sua concepção. Um nível de maturidade é composto por práticas específicas e genéricas relacionadas para um conjunto predefinido de áreas de processos que melhoram o desempenho global da organização. O nível de maturidade fornece uma maneira de caracterizar o seu desempenho da organização. Você sabe: quanto maior o nível de maturidade, mais organizada em seus processos a organização é. Em consequência, a tendência de gerar produtos de qualidade avançada é maior. Os níveis são o seguinte:

Nível 1 – Inicial: o processo de *software* é caracterizado como caótico. Poucos processos são definidos e o sucesso depende de esforços individuais. A organização não provê um ambiente estável para o desenvolvimento e manutenção do *software* e cronogramas e orçamentos são frequentemente abandonados por não serem baseados em estimativas próximas da realidade.

Nível 2 e Nível 3 – Gerenciado e Definido: as características dos níveis de maturidade 2 e 3 são semelhantes às dos níveis de capacidade 2 e 3.

Nível 4 – Quantitativamente gerenciado: nesse nível, a organização estabelece metas de qualidade e usa essas medidas na gestão de seus projetos. A qualidade de processo e de produto é entendida por meios estatísticos e gerenciada de forma que seja quantitativamente previsível.

Nível 5 – Em otimização: nesse nível, a organização melhora continuamente seus processos, com base nas medições quantitativas já obtidas (WAZLAWICK, 2013).

MPS.BR

O MPS.BR ou MR-MPS (Modelo de Referência para Melhoria do Processo de Software) é um modelo de avaliação voltado às empresas brasileiras, criado no ano de 2003 pela SOFTEX, em parceria com o Governo Federal e pesquisadores. Sua concepção leva em consideração normas e modelos internacionalmente reconhecidos, além das indispensáveis boas práticas da Engenharia de *Software* e a realidade de negócio dos desenvolvedores nacionais. Sua criação se justifica, inclusive, pelos altos custos de certificação nas normas internacionais.

O guia de implementação da norma se divide em 11 partes, sendo as 7 primeiras relacionadas aos 7 níveis de maturidade implantados no MPS.BR. As demais são guias de implementação nas empresas.

Esse modelo também é formado por níveis: Em otimização, Gerenciado Quantitativamente, Definido, Largamente Definido, Parcialmente Definido, Gerenciado, Parcialmente Gerenciado.

Em maio de 2015, haviam sido feitas 596 avaliações de *software* pela MPS.BR, por 13 instituições avaliadoras (WAZLAWICK, 2013). Disponível em: <http://www.softex.br/mpsbr/mps/mps-br-em-numeros/>. Acesso em: 24 jan. 2016.

Faça você mesmo

A norma ISO/IEC 15504, também conhecida como SPICE, foi criada para orientar a avaliação da capacidade das empresas nos processos. Trata-se de padrão bastante utilizado, que vale a pena ser conhecido. A prática aqui proposta é o levantamento e síntese, em relatório, da estrutura da SPICE e dos seus aspectos mais importantes.

SEM MEDO DE ERRAR

Por conta das atividades que você desenvolveu nas duas últimas seções, a X-Soft já conseguiu obter o registro das eventuais ações em favor da qualidade que a equipe tem executado e o relatório de avaliação de algumas características de um produto seu. O momento agora demanda empenho para a escolha e implantação de um modelo de qualidade do processo que seja economicamente viável, bem estruturado, reconhecido pelo mercado e que tenha boa adequação à cultura da X-Soft. Para que essa escolha seja feita com acerto, você deve recorrer ao conteúdo abordado na seção "Não pode faltar".

Conforme anteriormente colocado, o desafio inclui a criação de relatório contendo o processo de implantação de norma de qualidade na X-Soft, que vai desde os critérios para sua escolha, passa pelos objetivos da implantação e termina no planejamento dos passos que deverão ser tomados para sua efetiva adoção.

A indicação do conteúdo desse relatório vem na sequência.

1. O relatório deve ter início com a exposição dos objetivos da implantação de um modelo de qualidade.

2. O segundo item deve revelar a escolha do modelo e os motivos que levaram a ela. Questões como "a implantação consumirá muito recurso financeiro?", "trata-se de um modelo bem aceito no mercado?", "ele é adequado à realidade da nossa empresa?" deverão ser abordadas neste item.

3. Por fim, o relatório deverá identificar como o modelo será implementado. Aqui, as questões relacionadas à contratação de profissionais externos, disponibilização de treinamento externo à equipe, período de implantação e objetivos a serem alcançados deverão ser abordadas.

Atenção!

A ISO 90003 é uma versão mais atual da antiga norma ISO 9000-3:1997, que era um guia para aplicação da ISO 9001 a empresas de software. Um dos problemas com a 9000-3 é que ela não tratava a melhoria contínua do processo, mas apenas indicava os processos que as empresas deveriam manter. A 90003 corrigiu tal deficiência (WAZLAWICK, 2013).

Lembre-se

Você encontrará farta documentação sobre o modelo MPS.BR em: <http://www.softex.br/mpsbr/>. Acesso em: 25 jan. 2016. Essa documentação inclui guia geral do modelo para software e guias de avaliação, entre outros.

Avançando na prática

Pratique mais
Instrução Desafiamos você a praticar o que aprendeu transferindo seus conhecimentos para novas situações que pode encontrar no ambiente de trabalho. Realize as atividades e depois as compare com as de seus colegas.

Decidindo pelo padrão de qualidade adequado	
1. Competência Geral	Conhecer as principais metodologias de desenvolvimento de *software*, normas de qualidade e processos de teste de *software*.
2. Objetivos de aprendizagem	O objetivo desta seção é apresentar as normas de qualidade mais referenciadas.
3. Conteúdos relacionados	Introdução às normas de qualidade de *software*: ISO 9001 – Sistema de Gestão da Qualidade, ISO/IEC 90003 – Orientações para *software*, ISO/IEC 25010:2011 – Qualidade de Produto de *software* e CMMI – *Capability Maturity Model Integration*, MPS.BR.
4. Descrição da SP	Conforme relatado na última seção, a X-Soft tem experimentado bons níveis de excelência em seus produtos e, com isso, altos índices de satisfação de seus clientes. Há, no entanto, um novo desafio no horizonte: para ser aceita em licitação pública para desenvolvimento de *software* para o Governo Federal, a M-Soft deverá apresentar avaliação MPS-BR no ato da inscrição para o processo seletivo. Sua missão é relatar os passos a serem dados e especificar o órgão competente a ser consultado para que a avaliação seja obtida no menor tempo possível.
5. Resolução da SP	A solução do caso demanda as possíveis providências: 1. Um bom ponto de partida para a obtenção de informações gerais sobre a avaliação pode ser obtido em: <http://www.softex.br/a-softex/>, acesso em: 5 fev. 2016. Você encontrará nessa página informações sobre a SOFTEX, associação que busca promover a excelência do *software* brasileiro. 2. Na sequência, você poderá conseguir informações mais específicas sobre o processo em http://www.softex.br/mpsbr/guias/, acesso em 5 de fevereiro de 2016. Nessa página você encontrará Guia Geral de Serviços, Guia Geral de Software e o Guia de Avaliação, bases importantes para formar conhecimento sobre a avaliação. 3. Para complementar as consultas e obter mais informações a respeito, procure contato com empresa que já adota o MPS.BR. As avaliações realizadas desde setembro de 2005 podem ser encontradas em <http://www.softex.br/mpsbr/avaliacoes/mps-sw/>, acesso em: 25 jan. 2016.

Lembre-se

O processo de *software* pode ser relatado como um conjunto de ações, atividades, procedimentos e transformações utilizadas para construção e manutenção de *software*. (REZENDE, 2005).

Faça você mesmo

Faça o levantamento de qual o investimento real aproximado para a obtenção de avaliação CMMI e de uma avaliação MPS.BR.

Faça valer a pena!

1. Em relação a ISO, analise as afirmações que seguem:

I) Trata-se de organização governamental e atrelada à indústria norte-americana, responsável por emitir especificamente normas de software.

II) ISO significa International Software Orientation e trata-se de organização de destaque no ensino de criação de software.

III) A ISO é uma organização independente e sem vínculo com governos, que desenvolve e publica padrões internacionais para vários ramos de atividade.

IV) A ISO treina e certifica profissionais para o desenvolvimento de projetos de software.

É verdadeiro o que se afirma apenas em:

a) II e IV

b) II e III

c) III

d) IV

e) I e IV

2. Assinale a alternativa que contém apenas expressões diretamente relacionadas a norma ISO 9001.

a) Gestão de qualidade do produto e foco no cliente.

b) Baseada em 7 princípios de gerenciamento e destina-se à aplicação em qualquer organização.

c) Dependente do modelo de desenvolvimento de software e sem atualização desde sua criação.

d) Tomada de decisão baseada em evidências e gestão de qualidade do produto.

e) Criada pela ISO e mantida pela SOFTEX.

3. No contexto do CMMI, assinale a alternativa que contém expressões que completam corretamente as lacunas na frase abaixo.

"A representação _____ do CMMI usa níveis de maturidade para caracterizar o estado geral dos processos da organização em relação ao modelo como um todo, enquanto a representação _____ utiliza níveis de capacidade para caracterizar o estado dos processos da organização em relação a uma área de processo individual. O nível de capacidade _____ caracteriza-se por manter registros completos de descrição do processo. Estando no nível _____, uma organização já consegue conduzir processos monitorados, controlados e avaliados."

a) em estágios, contínua, definido, gerenciado

b) complementar, contínua, indefinido, automatizado

c) contínua, complementar, continuado, gerenciado

d) em estágios, complementar, gerenciado, definido

e) controlada, avançada, automatizado, definido

Seção 3.4

As verificações necessárias na Engenharia de *Software*

Diálogo aberto

Seja bem-vindo à última seção desta unidade!

Depois de diagnosticar como a qualidade era tratada pela equipe, avaliar seus principais indicadores em um produto e implementar o modelo padrão consagrado, a X-Soft mantém passo firme rumo à excelência em seus processos e na busca pela maior qualidade possível em seus produtos.

Como você está lembrado, o desafio da última seção foi justamente escolher o padrão de qualidade que a X-Soft adotaria. No entanto, você ainda pode incluir providências na rotina X-Soft que certamente implicarão retorno positivo à empresa. Que tal adotar formalmente um processo de detecção de defeitos rigorosos e bem definidos nos produtos? Será possível e viável quantificar algumas características do *software*, tal como complexidade funcional? Em outras palavras: conseguimos medir alguns atributos do programa? Entenda atributos como, por exemplo, número de funções e tamanho do código. Não é difícil imaginar o benefício que tais ações trariam aos produtos.

Pois bem, é hora de formalizar os procedimentos de qualidade do produto. Sua missão é justamente criar relatório contendo o processo de implantação de inspeções, medições e métricas de software. Dessa forma, poderá conhecer e conduzir processos de qualidade de *software*, aplicados a um projeto de desenvolvimento de *software*.

Conforme já tratamos, uma das consequências direta dessas ações é melhoria da qualidade do produto. Mas é certo que, indiretamente, ela é obtida pelo aprimoramento das estimativas de prazo, custo e esforço relacionadas ao produto. Estamos diante de algo a ser experimentado, não acha?

Bom trabalho e siga em frente com os estudos.

Não pode faltar

É muito difícil concebermos uma atividade desempenhada profissionalmente que não inclua maneiras de se medir o que é produzido. Essas medidas, sejam relacionadas a tamanho físico, a complexidade funcional ou a desempenho, visam assegurar algo indispensável em um ambiente produtivo: o controle. Nesta seção, você terá contato com o conceito de métrica e medição, com algumas formas de medições de um produto de *software*, além de revisões e inspeções. Com essas providências, a equipe deverá garantir e manter a qualidade de um *software*. Em frente!

Métrica e Medição

A medição é o processo pelo qual os números são atribuídos aos atributos de entidades do mundo real. Na medição, atribui-se um valor numérico a uma grandeza física. Por exemplo, quando medimos a distância entre dois pontos e obtemos 5 metros, estamos atribuindo o valor 5 à grandeza física chamada distância. Usaremos o termo medição tanto para descrever um processo como um valor de atributo. A **medição** é tipicamente uma quantificação direta, que envolve um único valor, ao passo que **métrica** é uma quantificação indireta, que envolve o cálculo e o uso de mais de uma medida. Vamos a um exemplo de métrica: considerando a medida de número de linhas de código de um programa e a medida de número de defeitos encontrados em todo o programa, podemos estabelecer a métrica de quantidade de defeitos por linha de código.

É desejável que as métricas sejam capazes de fornecer informação relevante para a tomada de decisão e para a comparação de desempenhos. É importante que você tenha em mente um fato importante: existem métricas referenciais, usadas normalmente de forma padronizada pelos desenvolvedores. No entanto, uma métrica pode ser baseada no objetivo da organização e na sua necessidade de informação para a tomada de uma decisão.

Devemos considerar também que uma métrica deve ser calculada com facilidade, que ela tenha condições de ser repetida quantas vezes forem necessárias, que sua unidade seja compreensível e universal e que, por fim, seu processamento possa ser automatizado.

Exemplificando

Observe este exemplo do uso de métrica. Se você quer comprar um carro – e procura por eficiência – você não levará em conta a quantidade de quilômetros que ele é capaz de rodar, nem a quantidade de combustível

que ele consegue armazenar no tanque. Na verdade, o que você quer saber é quanto ele consegue rodar com um litro de combustível. Assim, pelo uso de uma métrica, conseguimos chegar a um dado relevante e apto a te ajudar na tomada da sua decisão de compra do carro.

Imagine que dois projetos de *software* – com alguma similaridade funcional entre eles – estejam em construção. Se você quiser saber qual dos dois projetos está sendo mais produtivo, você deve realizar a **medição** do tamanho do projeto e do esforço para produzi-lo. Contudo, sem uma métrica, a comparação não seria viável, tampouco útil (MOORTHY, 2013).

No âmbito da construção de um *software*, as medidas podem ser categorizadas em (SWEBOK, 2004):

• **Medidas de Processo**: a expressão "medida de processo" que usamos aqui significa a coleta, análise e interpretação de informações quantitativas sobre o processo utilizado pelo desenvolvedor. A medição é usada para identificar os pontos fortes e deficiências do processo, além de avaliar o processo após sua implementação ou mudança. Por exemplo, a introdução de inspeções de *software* no processo pode reduzir o esforço para realização de testes. No entanto, pode haver aumento de tempo até a entrega do produto, caso as reuniões de inspeção sejam extensas demais. A decisão de se investir mais tempo nas inspeções do que nos testes, por exemplo, deverá ser tomada com base em métrica aplicada no processo.

• **Medidas de Produto**: as medidas de um produto incluem o tamanho do produto, sua estrutura e sua qualidade. Trataremos melhor dessas medidas e algumas das métricas correspondentes no decorrer desta seção. Outras duas medidas também devem ser consideradas neste contexto (MOORTHY, 2013):

• **Medidas de Projeto**: usadas para quantificar o desempenho de um projeto de *software*. Por exemplo, uma métrica comumente usada neste contexto é a porcentagem de variação no cronograma do projeto, assim expressa: (data real de término – data planejada de término *100) / (data planejada de término – data real de início).

• **Medidas de Recursos**: usadas para quantificar a utilização de recursos em um projeto de *software*. Imagine que estejamos tratando de recursos humanos, ou seja, pessoas. Uma boa medida de efetividade seria dada pela quantidade de tarefas cumpridas por unidade de tempo, considerando tarefas de complexidade e duração semelhantes.

Pesquise mais

Como uma organização pode definir qual conjunto de métricas é adequado aos seus objetivos? Existe um modelo que ajude a definir e implantar as métricas? O GQM, acrônimo para *Goal/Question/Metric* (Objetivo/Questão/Métrica) é uma abordagem orientada a metas para a mensuração de processos e produtos de *software*. Pesquise mais em:

<http://www.inf.ufsc.br/~gresse/download/cits99.pdf>. Acesso em: 12 fev. 2016.

<http://www.cin.ufpe.br/~scbs/metricas/seminarios/GQM_texto.pdf>. Acesso em: 12 fev. 2016.

WAZLAWICK, Raul Sidnei. **Engenharia de Software**: conceitos e práticas. Rio de Janeiro: Elsiever, 2013, p. 247.

Já sabemos que, feitas as medições, podemos (e devemos) utilizá-las para gerar as métricas. Para fins de classificação, algumas métricas são geradas a partir de medidas obtidas diretamente, geralmente por contagem do atributo observado. Às métricas geradas a partir dessas medidas damos o nome de **métricas diretas**. Outras métricas, porém, são obtidas indiretamente. A elas damos o nome de **métricas indireta**s.

Exemplificando

Exemplos de métricas diretas: custo, esforço, quantidade de linhas de código, quantidade de erros, velocidade de execução do programa, entre outras. Em relação às métricas indiretas, os exemplos mais referenciados são funcionalidade, confiabilidade e manutenibilidade.

Vale a pena analisarmos uma dessas métricas tomadas como exemplo. Na sequência, você conhecerá uma métrica de esforço, muito usada para medir o tamanho funcional de um *software* com o objetivo de obter boa estimativa de custo, antes mesmo da sua efetiva construção.

Análise de Pontos por Função

Essa técnica se baseia nos requisitos do *software* para a obtenção da métrica. Por isso, ela é aplicável a partir do momento em que os requisitos funcionais do programa (ou as histórias) tenham sido definidos. Esses requisitos (ou funções – daí o nome de Pontos por Função) são convertidos em valores numéricos que, depois de calculados e ajustados, proverão excelente ideia do esforço necessário para desenvolver o sistema (WAZLAWICK, 2013).

Você deve se lembrar de que uma medida direta se baseia geralmente em contagem feita em algum atributo do sistema. Os atributos relevantes neste nosso contexto são os requisitos.

Devemos considerar que os resultados obtidos pela aplicação dessa métrica não se prestam apenas para estimar esforço de desenvolvimento. Dependendo da intenção de uso da métrica, a contagem dos pontos de função também pode ser usada para:

• Melhoria de um produto – neste caso, são contadas as funcionalidades alteradas, suprimidas ou adicionadas durante a manutenção adaptativa.

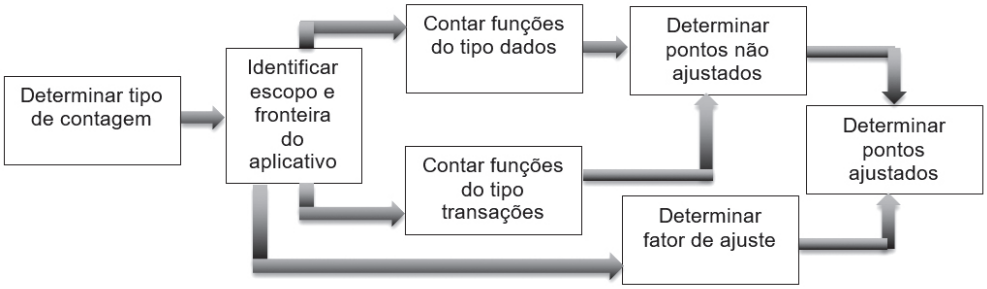
• Contagem de aplicação – técnica usada para contar pontos de aplicações já existentes.

Reflita

Quais as razões que justificam investimento de tempo e dinheiro para medir processos, produtos, projetos e recursos? Imagine que, por meio das medições, você poderá avaliar as atividades em curso, estimar em qual estágio tais atividades estarão no futuro e controlar o uso de recursos destinados ao projeto. Em outras palavras, as vantagens derivadas de um processo estruturado de medições poderão oferecer claras indicações sobre a qualidade do produto, servirão de avaliação da produtividade da equipe e determinarão se um novo método ou ferramenta implementados estão sendo efetivos.

Pois bem, falta-nos então conhecer o procedimento de contagem das funções e do cálculo da métrica. Antes de você chegar a qualquer explicação textual, observe a figura 3.2:

Figura 3.2 – Diagrama do processo de contagem de pontos de função



Fonte: MECENAS; OLIVEIRA, 2005, p. 4.

A fase de identificação da fronteira do aplicativo serve, por exemplo, para determinar se a contagem estará concentrada em um ou mais sistemas. Ela serve para estabelecer um divisor entre os componentes do aplicativo e os componentes de outro aplicativo.

Na sequência do processo, vem a contagem das funções do sistema. Vejamos o procedimento (MECENAS; OLIVEIRA, 2005):

• **Funções do tipo dados**: representam as necessidades de dados dos usuários, de acordo com sua visão de negócio. Divide-se em:

Arquivo Lógico Interno (ALI): trata-se de um elemento percebido pelo usuário e mantido internamente pelo sistema. Exemplos: arquivos de cadastro de clientes, cadastro de funcionários, arquivos de mensagens de auxílio e arquivos de mensagens de erros.

Arquivo de Interface Externa (AIE): representa as necessidades de dados externos à aplicação, ou seja, são dados armazenados fora da fronteira da aplicação, mas que não sofrem manutenção internamente.

• **Funções do tipo transação**: representam as funcionalidades de processamento de dados identificados pelos usuários. Existem três funções deste tipo:

Entrada Externa (EE): função que obtém dados informados pelo usuário ou por outra aplicação e os inserem no sistema. A função deve ter como objetivo armazenar, alterar ou remover dados no sistema. O nome de um cliente e seu endereço são exemplos de entradas externas.

Saída Externa (SE): função que obtém dados do sistema e apresentam ao cliente ou enviam a outras aplicações, sendo que pelo menos um valor obtido por cálculo deve existir para que seja considerada saída externa. Exemplo: fatura de um cliente, relação de clientes inadimplentes.

Consulta Externa (CE): função que apresenta dados da mesma forma que foram armazenados, sem cálculos ou transformações. Configura uma consulta externa, por exemplo, informar o CPF de uma pessoa ao sistema para se obter seus dados cadastrais completos.

Uma vez identificadas e contadas as funções, as quantidades apuradas são classificadas como complexidade alta, média e baixa, conforme a tabela 3.3:

Tabela 3.3 – Fatores de complexidade

	Complexidade Funcional		
Tipo de função	Baixa	Média	Alta
Entrada	3	4	6
Saída	4	5	7
Consulta	3	4	6
Arquivo interno	7	10	15
Arquivo externo	5	7	10

Fonte: WASLAWICK, 2013, p. 161.

Assimile

Para efeito de contagem, um requisito do sistema equivale a uma função. No entanto, essa regra não deve ser tomada de forma absoluta. Apenas funções visíveis para o usuário devem ser consideradas. Um cálculo interno, por exemplo, não deve ser contado. Se apenas um requisito trata de cadastro de clientes e de produtos, então teremos aí duas funções. Em resumo, deve-se tomar os requisitos, eliminar os que são funções internas e subdividir aqueles que representam mais do que uma função (WAZLAWICK, 2013).

Determinadas as complexidades das funções do tipo dados e transação, a soma dos pontos obtidos é chamada de pontos de função não ajustado. Nessa etapa, já temos o tamanho funcional do aplicativo (MECENAS; OLIVEIRA, 2005).

Ocorre que, para que essa métrica possa ser útil de fato, há que se levar em consideração a real complexidade técnica dessas funções. Por exemplo, um sistema de controle de uma loja de locação de vídeo tende a possuir funções mais simples e estruturadas do que um sistema bancário. O método, então, prevê a aplicação de ajuste neste valor obtido em função de 14 características gerais do sistema, aplicáveis a todo o projeto. Seguem:

1) Comunicação de dados: em que grau a comunicação de dados é requerida?

2) Processamento de dados distribuído: em que grau o processamento distribuído está presente?

3) *Performance*: o desempenho é fator crítico na aplicação?

4) Uso do sistema: o usuário deseja executar a aplicação em um equipamento já existente ou comprado e que será altamente utilizado?

5) Taxa de transações: qual o volume de transações esperado?

6) Entrada de dados *on-line*: são requeridas entrada de dados *on-line*?

7) Eficiência do usuário final: as funções interativas fornecidas pela aplicação enfatizam um projeto para o aumento da eficiência do usuário final?

8) Atualização *on-line*: há arquivos atualizados *on-line*?

9) Processamento complexo: qual o grau de complexidade do processamento interno?

10) Reusabilidade: em que grau o código é reutilizável?

11) Facilidade de instalação: em que grau o sistema é fácil de ser instalado?

12) Facilidade de operação: em que grau o sistema é fácil de ser operado?

13) Múltiplos locais: o sistema é projetado para múltiplas instalações em diferentes organizações?

14) Facilidade para mudanças: a aplicação é projetada de forma a facilitar mudanças?

Como cada um receberá uma nota de 0 a 5, o somatório desses fatores ficará entre 0 e 70. Assim, finalmente, a fórmula para o cálculo dos pontos por função ficará como segue:

$$VAF = 0{,}65 + \left(0{,}01 * \sum_{i=1}^{14} GSCi\right)$$

Onde: VAF = *Value Adjustment Factor* ou Fator de Ajuste da Função;

GSC = *General Systems Characteristics* ou Características Gerais do Sistema.

Por fim, tendo sido obtido os pontos não ajustados, que chamaremos de UFP (*Unadjusted Function Points*, ou pontos por função não ajustados), basta aplicar a fórmula que segue para o número de pontos por função ajustado:

$$AFP = UFP\ x\ VAF$$

Vamos a um exemplo simples de cálculo de pontos por função. Suponha um projeto de programa que gerou as seguintes contagens:

Tabela 3.4 – Cálculo de Pontos por função

Componente	Quantidade contada	Complexidade Funcional	Fator de complexidade	Total de complexidade	Total do componente
Arquivo Lógico Interno	3	Baixa	7	21	21
	0	Média	10	0	
	0	Alta	15	0	
Arquivo de Interface Externa (AIE)	2	Baixa	5	10	10
	0	Média	7	0	
	0	Alta	10	0	
Entradas Externas (EE)	4	Baixa	3	12	26
	2	Média	4	8	
	1	Alta	6	6	
Saídas Externas (SE)	2	Baixa	4	8	8
	0	Média	5	0	
	0	Alta	7	0	
Consultas Externas (CE)	2	Baixa	3	6	14
	2	Média	4	8	
	0	Alta	6	0	

Fonte: Wazlawick (2013, p. 161).

Teremos então o valor 79 como contagem dos pontos por função não ajustados. Supondo que tenhamos obtido 55 na soma das características gerais do sistema, a fórmula de cálculo do Fator de Ajuste da Função ficaria como segue:

VAF=0,65+0,55

VAF=1,2

Ao aplicarmos a fórmula dos pontos ajustados, obteremos: *AFP*=79*1,2. Portanto, os pontos por função ajustados para este caso é 94,8.

Revisões e inspeções de *software*

Como você bem se recorda, revisões técnicas e inspeções foram apresentados na primeira seção desta unidade. Dada a importância de ambos e a intenção de introduzir

desde já temas relacionados a teste de *software*, uma nova abordagem será feita aqui. Uma inspeção típica apresenta cinco etapas formais (SCHACH, 2008):

1. Uma visão geral do documento a ser inspecionado (histórias, projeto, código ou outro) é dada e, logo após, é distribuído aos presentes na sessão;

2. Por meio de uma lista de imperfeições detectadas em inspeções recentes, a equipe deverá analisar e entender o presente documento;

3. Um participante escolhido deve percorrer o documento e conduzir a equipe de inspeção, garantindo que cada item seja verificado, em busca de falhas. A missão é encontrar falhas, não as corrigir. No prazo de um dia, o líder da equipe dever gerar relatório do que foi discutido;

4. No processo chamado **reformulação**, o responsável resolve as imperfeições encontradas;

5. No **acompanhamento**, o responsável deve garantir que cada questão levantada tenha sido adequadamente resolvida.

Para determinar a eficácia de uma inspeção, você pode usar a métrica da **taxa de inspeção**. Quando, por exemplo, a especificação dos requisitos (ou histórias) passam por inspeção, o número de páginas inspecionadas por hora pode ser medido. Se a inspeção recair sobre o código, uma métrica interessante é a de quantidade de linhas de código inspecionadas por hora.

Lembre-se

Formalmente, Inspeção e *Walkthroughs* (passo a passo) são processos diferentes. Este último é um processo de duas etapas apenas: uma de preparação e outra de análise de documento pela equipe.

Faça você mesmo

Considere um sistema que você conheça bem e separe, classifique e pontue suas funções de acordo com o estipulado para o cálculo de pontos por função. Na sequência, aplique os fatores de ajuste e obtenha o valor ajustado das funções, conforme a fórmula dada.

No próximo item será aplicado o conteúdo que acabamos de abordar com o objetivo de resolver o desafio proposto.

SEM MEDO DE ERRAR

A busca da X-Soft pela excelência em seus processos e produtos continua. Com a sua intervenção, a empresa expandiu o conceito de qualidade e agora mira nas medições dos atributos de seus programas para obter controle sobre aquilo que produz. A implantação de medições e as métricas que delas derivam servirão para inspirar ainda mais a equipe na demanda pela qualidade.

No item anterior a este, você teve a oportunidade de absorver um pouco do conteúdo relacionado a métrica de **Pontos por Funçã**o, juntamente com os conceitos de medição e métrica de *software*. Tais conhecimentos, aliados à sua crescente habilidade em planejar implantações de novos procedimentos, permitirá a você superar o desafio que ora é proposto. Conforme já discutido, você deverá relatar seu planejamento para implantar processo de obtenção de métricas de *software* no cotidiano da X-Soft. Vamos então a uma possível solução para o desafio, utilizando a métrica de Pontos por Função como objeto.

1. Você deverá definir que o resultado obtido pela aplicação da métrica de Pontos por Função servirá para estimar o esforço para criação dos novos produtos. Em outras palavras, o procedimento deverá ser aplicado antes que o produto fique pronto.

2. A equipe responsável pela coleta e tratamento das histórias ou dos requisitos deverá ser orientada para que prepare as funções para serem processadas, excluindo cálculos internos da contagem e dividindo mais de uma função no mesmo contexto em duas funções distintas.

3. A equipe responsável pela aplicação da métrica deverá, então, contar as funções, obter os pontos por função não ajustados. Com utilização de critérios próprios e baseados na natureza de cada projeto, a equipe deverá estipular os valores que serão aplicados a cada um dos 14 fatores de ajuste, a fim de obter os pontos por função ajustados.

4. Após sua obtenção, o resultado deverá ser encaminhado para avaliação e análise, de forma que o esforço para criação daquele programa seja quantificado.

Com esse planejamento e com o bom aproveitamento do conteúdo ministrado, você inicia sua preparação para gerenciar processos de aplicações de medições e métricas em sua vida profissional. Dependendo de certas demandas, você poderá também incluir a aplicação de outras métricas no cotidiano da X-Soft.

Atenção!

A aplicação dos fatores de ajuste nos pontos obtidos não é procedimento obrigatório. Em, por exemplo, uma organização que produz programas com graus de dificuldade homogêneos, a prática poderá ser dispensada.

Lembre-se

A métrica de Pontos por Função não se aplica apenas aos programas em fase de planejamento. Ela poderá ser aplicada também em processos de manutenção, permitindo a estimativa do esforço necessário para se fazer certa alteração no programa.

Avançando na prática

Pratique mais
Instrução Desafiamos você a praticar o que aprendeu transferindo seus conhecimentos para novas situações que pode encontrar no ambiente de trabalho. Realize as atividades e depois as compare com as de seus colegas.

Fazendo estimativas com Pontos por Função	
1. Competência Geral	Competência para conhecer e conduzir processos de qualidade de *software*.
2. Objetivos de aprendizagem	Introduzir métricas e inspeções de software.
3. Conteúdos relacionados	Introdução a revisões, inspeções, medições e métricas.
4. Descrição da SP	A empresa X-Soft, devidamente estabelecida no mercado e que vem desenvolvendo cultura de qualidade e de aprimoramento constante de seus produtos ao longo do tempo, também implantou recentemente a métrica de Pontos por Função. No entanto, embora a contagem das funções e o cálculo estejam sendo feitos corretamente, os resultados não têm servido para estimar o esforço efetivo a ser empreendido em seus projetos de criação de programas. Ou seja, o resultado é obtido, mas não é adequadamente utilizado. Sua missão é sugerir um procedimento que permita que a X-Soft seja capaz de estimar o esforço em um projeto, com base no resultado obtido pela aplicação da métrica Pontos por Função.
5. Resolução da SP	O esforço total de construção do programa é dado pela multiplicação do Índice de Produtividade (IP) pelo resultado obtido pela aplicação da métrica Pontos por Função (APF). A fórmula então fica: *E = APF * IP*

	Naturalmente que a variável mais suscetível a grandes variações é justamente o índice de produtividade, que varia em função da experiência da equipe, do ambiente de trabalho e da aptidão da equipe para aquele projeto em específico. Um dos cálculos possíveis para a obtenção do Índice de Produtividade se perfaz pela tomada de um projeto anteriormente desenvolvido, com esforço conhecido como base.

Lembre-se

Medidas e métricas serão valores frios e sem utilidade se não interpretados adequadamente.

Faça você mesmo

Pesquise sobre e resuma os conceitos de Arquivo Lógico Interno (ALI) e Arquivo de Interface Externa (AIE); Entrada Externa (EE), Saída Externa (SE) e Consulta Externa (CE). Inicie sua pesquisa por: <http://www.inf.ufpr.br/silvia/ES/metricas/FP.pdf>. Acesso em: 12 fev. 2016.

Faça valer a pena!

1. Assinale a alternativa que descreve o conceito de métrica.

a) Processo pelo qual os números são atribuídos aos atributos de entidades do mundo real.

b) Quantificação direta, que envolve um único valor.

c) Quantificação indireta, que envolve o cálculo e o uso de mais de uma medida.

d) Escala de valores que dá a noção de uma medida de um determinado elemento.

e) Processo pelo qual se avalia a capacidade de cada desenvolvedor com base em seu tempo de experiência na função.

2. Em relação a revisões de software, assinale a afirmação verdadeira:

a) Uma revisão típica apresenta duas etapas formais e, por isso, equivale ao walkthrough.

b) Numa das etapas da revisão, o responsável pela reunião faz a refatoração do código defeituoso.

c) É na etapa conhecida como acompanhamento que o responsável pela revisão deve garantir que cada questão levantada tenha sido adequadamente resolvida.

d) Por não ser tangível, não há maneiras de se medir um procedimento de revisão.

e) A revisão consiste apenas na leitura cuidadosa do código-fonte, linha a linha, em busca de defeitos.

3. Em relação ao processo de Análise de Pontos por Função, analise as afirmações que seguem:

I) Trata-se de medida de dificuldade em se programar determinadas funções do programa.

II) Os requisitos são convertidos em valores numéricos que, depois de calculados e ajustados, fornecerão estimativa do esforço necessário para desenvolver o sistema.

III) Aplicável a partir do momento em que os requisitos funcionais do programa (ou as histórias) tenham sido definidos.

IV) Os resultados obtidos pela aplicação dessa métrica valem apenas para estimar esforço de desenvolvimento.

É verdadeiro o que se afirma apenas em:

a) IV

b) II e IV

c) III e IV

d) II e III

e) I

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.

BARTIÉ, A. **Garantia da qualidade de software**: As melhores práticas de Engenharia de Software aplicadas à sua empresa. 5. ed. São Paulo: Elsiever, 2002.

CMMI INSTITUTE. **Who uses CMMI**? Disponível em: <http://cmmiinstitute.com/who-uses-cmmi>. Acesso em: 24 jan. 2016.

CROSBY, P. B. **Quality is free**: the art of making quality certain. New York: New American Library, 1979.

DENNIS, R. G; DIANE, L. G. **Demonstrating the impact and benefits of CMMI**: an update and preliminary results. 2003. Disponível em: <http://resources.sei.cmu.edu/asset_files/SpecialReport/2003_003_001_14117.pdf>. Acesso em: 5 fev. 2016.

DEVMEDIA. **CMMI**: uma visão geral. Disponível em: <http://www.devmedia.com.br/cmmi-uma-visao-geral/25425#ixzz3yBwjcLs1>. Acesso em: 24 jan. 2016.

GOVERNO ELETRÔNICO. **ePING – Padrões de interoperabilidade de Governo Eletrônico**. Disponível em: <http://www.governoeletronico.gov.br/acoes-e-projetos/e-ping-padroes-de-interoperabilidade/o-que-e-interoperabilidade>. Acesso em: 18 fev. 2016.

IEEE (Instituto de Engenheiros Eletricistas e Eletrônicos, do inglês *Institute of Electrical and Electronics Engineers*). **SWEBOK *(Guide to the Software Engineering Body of Knowledge)***. 2004. Disponível em: <https://www.computer.org/portal/web/swebok>. Acesso em: 22 fev. 2016.

HUMPHREY, W., KITSON, D., KASSE, T. The state of software engineering practice: a preliminary report. In: **Proceedings of the 11th International Conference on Software Engineering**. p. 277-288, 1989.

IME. **Ariane 5**: um erro numérico (*overflow*) levou à falha no primeiro lançamento. Disponível em: <http://www.ime.uerj.br/~demoura/Especializ/Ariane/>. Acesso em: 17 jan. 2015.

ISO. **About ISO**. Disponível em: <http://www.iso.org/iso/home/about.htm>. Acesso em: 14 jan. 2016.

LEÃO, A. A**plicação de técnicas de ITIL e CMMI em pequenas e médias empresas de software**. Disponível em: <http://www.techoje.com.br/site/techoje/categoria/detalhe_artigo/1734>. Acesso em: 5 fev. 2016.

MECENAS, I.; OLIVEIRA, V. **Qualidade em Software**: Uma metodologia para homologação de sistemas. [s. l.]: Alta Books, 2005.

MOORTHY, V. **Jumpstart to Software Quality Assuranc**e. [s.l : s.n.], 2013.

PORTAL EDUCAÇÃO. **A história da organização ISO**. Disponível em: <http://www.portaleducacao.com.br/administracao/artigos/40732/a-historia-da-organizacao-iso#!1>. Acesso em: 28 jan. 2016.

PRESSMAN, R. S. **Engenharia de software**. São Paulo: Pearson Prentice Hall, 2009. 1056 p.

PRESSMAN, R. S. **Engenharia de software**. São Paulo: Makron Books,1995.

QUALITY CONSULT. **O que muda na versão 2015 da norma ISO 9001**. Disponível em: <http://www.qualityconsult.com.br/index.php/iso-9001-versao-2015/>. Acesso em: 23 jan. 2016.

REISS, Eric. **Usable usability**: simple steps for making stuff better. Indianapolis: John Wiley & Sons, 2012.

REZENDE, Denis Alcides. **Engenharia de software e sistemas de informação**. 3. ed., rev. Rio de Janeiro: Brasport, 2005.

ROCHA, A. R.; MALDONADO, J. C.; WEBER, K. C. **Qualidade de softwar**e: Teoria e Prática. São Paulo: Pearson, 2001.

RODRIGUES, A. N. et al. **Qualidade de Software – parte 1**. Disponível em: <http://www.devmedia.com.br/qualidade-de-software-parte-01/9408>. Acesso em: 11 jan. 2016.

SCHACH, S. **Engenharia de Software**: Os Paradigmas Clássico e Orientado a Objetos. 7. ed. São Paulo: McGraw-Hill, 2008.

SHAFFER, S. C. **A Brief Introduction to Software Development and Quality Assurance Management**. [S.l.: s.n.], 2013.

SEEAR, David Jonh. **ISO 9001: 2015 Back to the future**. A review of the new ISO Annex SL structure for Certification Standards using the draft ISSO 9001: 2015 to explain the changes. USA: Author House, 2015.

SILVEIRA, S. G. **Carreira: Analista de Teste – O que é e o que faz?** Disponível em: <http://www.tiegestao.com.br/2013/09/01/carreira-analista-de-teste-o-que-e-e-o-que-faz/>. Acesso em: 20 jan. 2016.

SOFTEX. **MPS.BR**. Disponível em: <http://www.softex.br/mpsbr/mps/mps-br-em-numeros/>. Acesso em: 24 jan. 2016.

REISS, Eric. **Usable Usability**: Simple Steps for Making Stuff Better. [S.l]: Wiley, 2012. 256 p.

TSUKUMO, A. N. *et al*. **Qualidade de Software**: Visões de Produto e Processo de Software. In: II Escola Regional de Informática da Sociedade Brasileira de Computação Regional de São Paulo – II ERI da SBC – Piracicaba, SP – junho de 1997, p. 173-189. Disponível em: <https://xa.yimg.com/kq/groups/21646421/371309618/name/Modelos+de+Qualidade+de+Software.pdf>. Acesso em: 18 jan. 2016.

WAZLAWICK, R. S. **Engenharia de Software**: conc