Qualidade de Software
O que é qualidade?

Herysson R. Figueiredo
herysson.figueiredo@ufn.edu.br

Sumário

- História
- Qualidade e Requisitos
- Subjetividade
- Qualidade e Bugs
- Defeito, falha ou bug
- Qualidade e o SWEBOK

História

Um grande marco na história da qualidade foi, com certeza, a revolução industrial (1750-1840). Esse período também é associado a profundas mudanças econômicas e sociais, como o início da automação e o surgimento do consumo de massa.

História

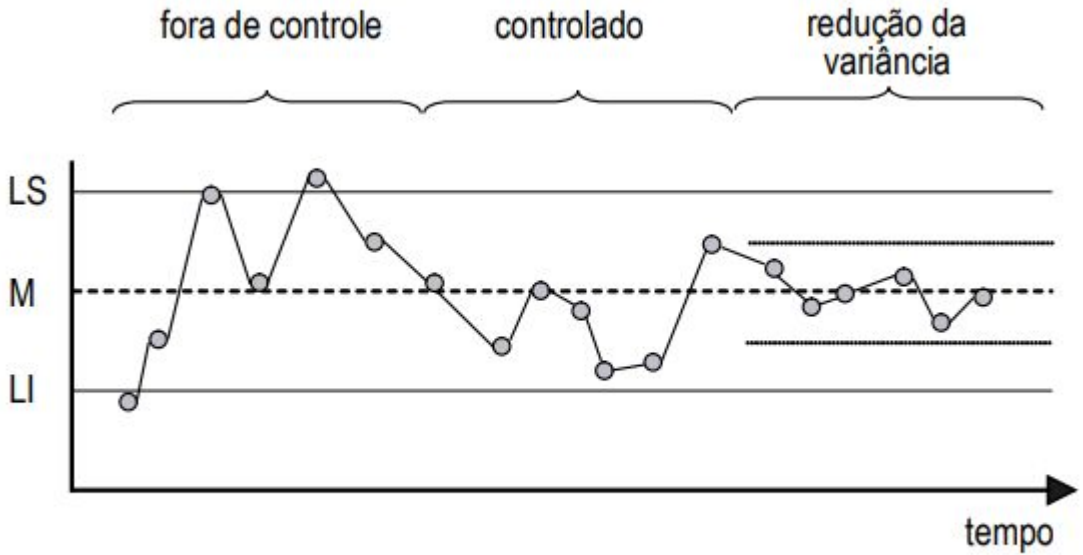
Na década de 1920 surgiu o controle estatístico de produção. Nas fábricas que produziam grande quantidade de itens tornou-se impossível garantir a qualidade individual de cada peça, ao contrário do que se fazia (e ainda se faz) no trabalho artesanal.

História

Um dos primeiros trabalhos associados ao assunto é o livro publicado por Walter Shewhart em 1931, *Economic Control of Quality of Manufactured Product*. Shewhart, dos Bell Laboratories, teria introduzido os diagramas de controle (*control charts ou Shewhart chart*).

História

Um d publicado
por Walte *nufactured*
Product. Sl gramas de
controle (



LS = Limite superior; M = Média; LI = Limite inferior.

História

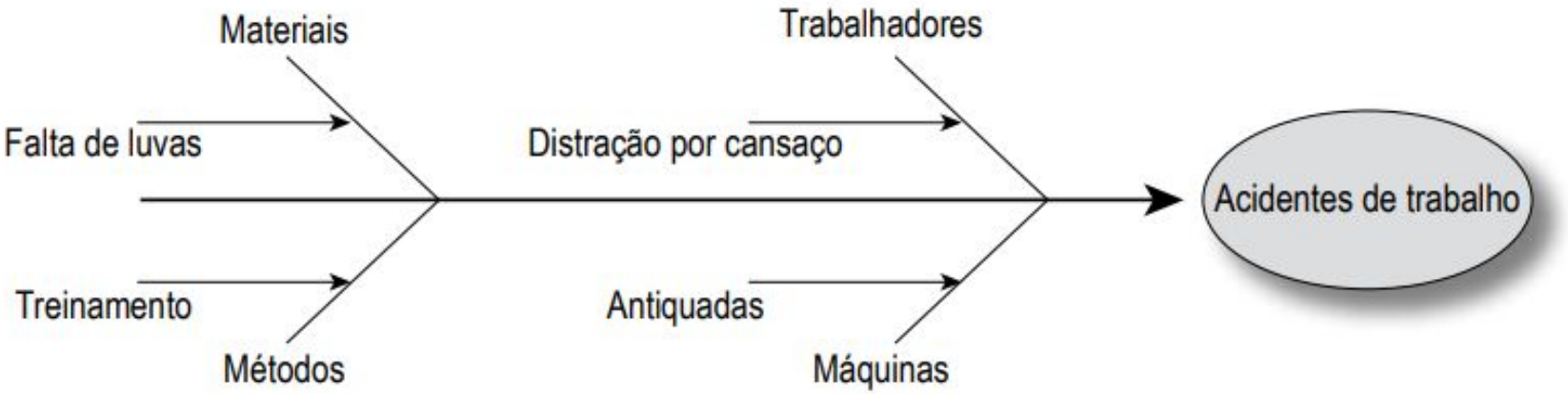
Na década de 1940 surgiram vários organismos ligados à qualidade; por exemplo, a ASQC (*American Society for Quality Control*), a ABNT (Associação Brasileira de Normas Técnicas) e, ainda, a ISO (International Standardization Organization). A Segunda Guerra Mundial também contribuiu com o processo, quando as técnicas de manufatura foram aprimoradas para fabricação de material bélico.

História

Na década de 1940 o Japão destacou-se como um importante pólo no assunto e contribuiu com diversas novas ferramentas: o método de Taguchi para projeto experimental, a metodologia 5S ou, ainda, os diagramas de causa e efeito de Ishikawa, também conhecidos como diagramas espinha de peixe.


SEIRI
Senso de Utilização.
Combater distrações que atrapalhem tanto a concentração quanto a produtividade de todos no ambiente corporativo.
5S
SEITON
Senso de Organização.
Organização é o que determina o segundo S da metodologia 5S. Por isso, mesas desorganizadas, ou arquivos espalhados não fazem o menor sentido.
SEISO
Senso de Limpeza.
Diz respeito àquilo que todos os colaboradores precisam ter. Tanto com os seus instrumentos de trabalho como com o local onde realiza suas atividades.
SEIKETSU
Senso de Padronização e Saúde.
Criar processos que garantam a aplicação dos S anteriormente citados.
SHITSUKE
Senso de Disciplina e Autodisciplina.
Sentimentos gerados pela mudança através de toda a metodologia 5S.

História


Materiais
Trabalhadores
Falta de luvas
Distração por cansaço
Acidentes de trabalho
Treinamento
Antiquadas
Métodos
Máquinas

Crise de mais de 30 anos

A maior causa da crise do software é que as máquinas tornaram-se várias ordens de magnitude mais potentes! Em termos diretos, enquanto não havia máquinas, programar não era um problema; quando tivemos computadores fracos, isso se tornou um problema pequeno e agora que temos computadores gigantescos, programar tornou-se um problema gigantesco. (meados de 1970).

Crise de mais de 30 anos

Segundo relatório da conferência da NATO de 1968 e outros documentos produzidos na década de 1970, fazemos uma descoberta assustadora: os problemas são os mesmos que encontramos atualmente.

Crise de mais de 30 anos

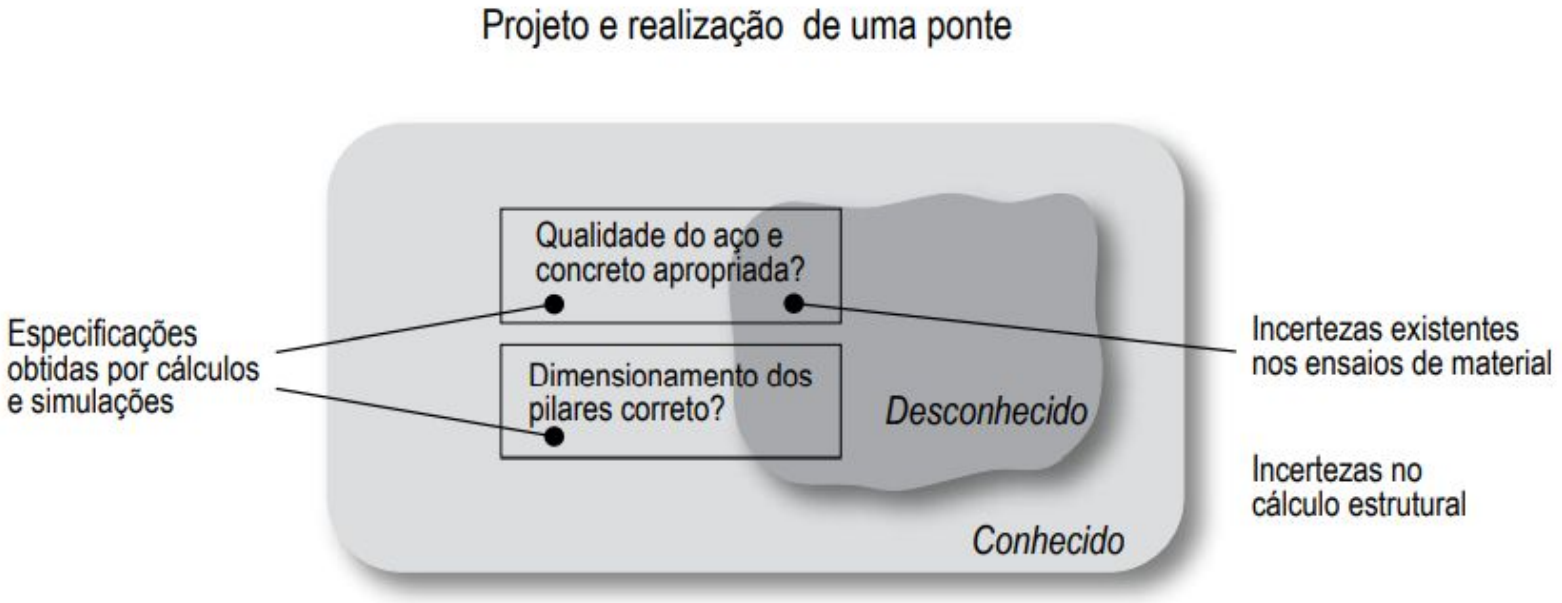
Façamos uma pequena lista:

- cronogramas não observados;
- projetos com tantas dificuldades que são abandonados;
- módulos que não operam corretamente quando combinados;
- programas que não fazem exatamente o que era esperado;
- programas tão difíceis de usar que são descartados;
- programas que simplesmente param de funcionar

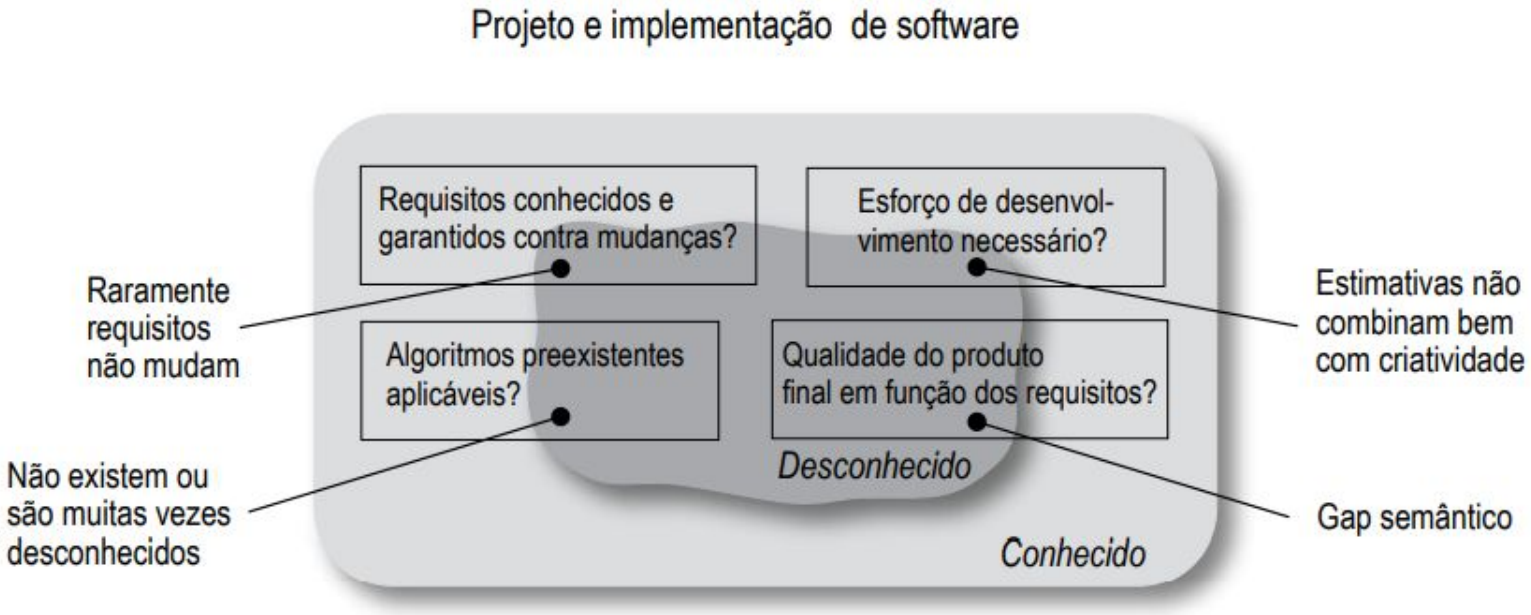
Crise de mais de 30 anos

O aspecto não repetitivo do desenvolvimento de software torna essa atividade difícil e, sobretudo, em boa medida imprevisível. Apenas uma pequena parcela da construção de software corresponde a atividades que poderíamos chamar de "montagem”

Crise de mais de 30 anos



Crise de mais de 30 anos



Qualidade e Requisitos

Uma das primeiras questões a responder quando o assunto é qualidade é como julgá-la. Por exemplo: se estamos diante de produtos alternativos, como escolher o melhor?

Qualidade e Requisitos

Esse problema de julgamento acontece com qualquer pessoa cotidianamente, quando se consomem itens como roupas, música, comida ou filmes. Mas curiosamente, apesar da frequência com que avaliamos os objetos à nossa volta, é muito difícil obter consenso a respeito da qualidade de um produto.

Qualidade e Requisitos

Uma escolha torna-se mais clara quando se estabelecem critérios que sirvam para julgar um produto. Em algumas situações, tais critérios são relativamente simples de identificar e estabelecer.

Qualidade e Requisitos

Crosby [1992]: "A qualidade é conformidade aos requisitos".

$$qualidade = f(requisitos)$$

Há, contudo, três fatos que perturbam essa definição, os quais é preciso conhecer para poder aplicá-la corretamente.

Qualidade e Requisitos

Em primeiro lugar, a definição nos deixa com a tarefa de definir o que é conformidade. Em alguns casos, isso se traduz em uma decisão booleana: uma lâmpada de 60W não é uma lâmpada de 100W. Mas, em geral, há poucos requisitos que possam ser tratados dessa maneira.

Qualidade e Requisitos

O que se faz, então, é especificar margens de precisão: uma lâmpada que consuma 59,9W é melhor que outra consumindo 60,1W. Esse exemplo nos leva naturalmente a considerar que possam existir intensidades ou graus de qualidade. Podemos escrever isso assim:

$$qualidade = f(observado, especificado)$$
$$= \|observado - especificado\|$$

Qualidade e Requisitos

O segundo ponto diz respeito à realização da observação do produto. Como sabemos que a lâmpada consome exatamente 60,1W? Não seriam talvez 60,2W?

Existem várias fontes de erro que podem corromper os dados utilizados para caracterizar um produto.

O erro de observação pode ser representado assim:

$$qualidade = \|observado - especificado + \varepsilon\|$$

Qualidade e Requisitos

Por fim, o terceiro ponto a considerar é o papel de diferentes clientes em um mesmo projeto. Uma ótima exposição do assunto é feita por Weinberg [1994]: os requisitos foram definidos por alguém, logo a qualidade depende das escolhas que alguém efetuou.

Papel da subjetividade

A qualidade de um produto tem um propósito: satisfazer o cliente. Esse objetivo implica tratar um domínio, em geral, bastante nebuloso.

Papel da subjetividade

Para compreender o motivo, considere o caso de uma pessoa que deve adquirir um produto comum no mercado. Ninguém compra uma camisa pensando nas propriedades mecânicas do tecido com o qual ela foi fabricada: “Que bela camisa! Pode resistir a 10 kg de tração!”. Em vez disso, são fatores muito difíceis de medir que, em geral, terão maior peso na decisão

Qualidade e bugs

Geralmente o simples fato de pronunciar a palavra bug equivale a acionar um alarme e fazer uma equipe de programadores estressados entrar em pânico. A discussão sobre o assunto se resume tipicamente à equação inexata

$$qualidade = \overline{bug}$$

Qualidade e bugs

O dilema gerencial: este caso é narrado por Weinberg [1994]: é a história de um programa de edição de texto.

Qualidade e bugs

A importância relativa I: muitos programas de computador possuem defeitos conhecidos e considerados pelos usuários como de menor importância.

Qualidade e bugs

A importância relativa II: o sistema de tipografia TeX é lendário pela qualidade de resultado impresso, por ser gratuito e pelo fato de que, depois de anos, poucos erros terem sido encontrados.

Qualidade e bugs

A qualidade de um software, como se pode ver, depende de se decidir o que significa qualidade! Não é um assunto que possa ser tratado com dogmas: “Não cometerás erros de programação”.

Qualidade e bugs

Em vez disso, é preciso adotar uma perspectiva técnica e considerar diversos fatores que afetam a construção do produto e que influenciem no julgamento dos usuários:

- tamanho e complexidade do software sendo construído;
- número de pessoas envolvidas no projeto;
- ferramentas utilizadas;
- custos associados à existência de erros;
- custos associados à detecção e à remoção de erros.

Um erro é um defeito, uma falha ou bug?

Qual a melhor palavra para explicar que um programa “travou” ou não funciona corretamente?

Um erro é um defeito, uma falha ou bug?

Há termos relacionados com erros de programação que, algumas vezes, provocam um pouco de confusão. Embora evoquem idéias parecidas, defeito, erro e falha não são sinônimos entre si e são usadas para designar conceitos distintos.

Um erro é um defeito, uma falha ou bug?

Defeito: é uma imperfeição de um produto. O defeito faz parte do produto e, em geral, refere-se a algo que está implementado no código de maneira incorreta.

Um erro é um defeito, uma falha ou bug?

Mas a palavra “defeito” não significa apenas um problema que faz um programa não funcionar. Um programa defeituoso, segundo o dicionário Houaiss, é um programa “que não funciona como deve”.

Um erro é um defeito, uma falha ou bug?

O código, em C++, faz o seguinte:

atribui 10.000.000.000 à variável a;

soma 0.1 a esse valor e o armazena em b;

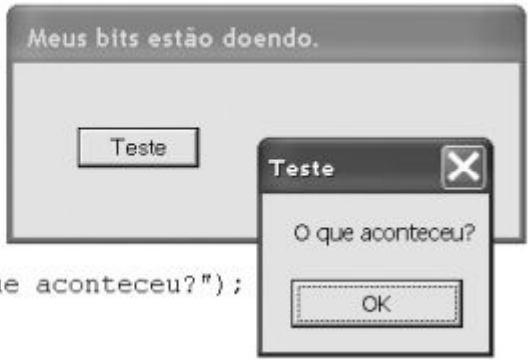
subtrai 10.000.000.000 desse valor.

```
int teste (void) {

   float a,b,c,d;


   a = 1e10;
   b = a + 0.1;
   b = b - a;

   if (0 == b) {
      shout ("O que aconteceu?");
   };
```



Um erro é um defeito, uma falha ou bug?

Falha é o resultado errado provocado por um defeito ou condição inesperada.

Um erro é um defeito, uma falha ou bug?

Os defeitos podem existir, mas nem sempre ser visíveis. Falhas também podem ocorrer por fatores externos ao programa, como corrupção de bases de dados ou invasões de memória por outros programas

Um erro é um defeito, uma falha ou bug?

Como foi dito antes, as falhas que chamam mais a atenção são certamente aquelas em que o programa trava. Contudo, toda falha potencial pode ser perigosa, mesmo se o programa não for paralisado.

Um er

Como foi amente
aquelas e ode ser
perigosa,

ENGANO (*MISTAKE*)	Uma ação humana que produz um resultado incorreto, como uma codificação ou modelagem errada;
DEFEITO (*FAULT*)	Uma imperfeição ou deficiência em um artefato que faz com que este não esteja em conformidade com os requisitos ou especificações, sendo necessária sua correção ou substituição. Termos como "erro" e "*bug*" comumente são usados para expressar defeitos;
ERRO (*ERROR*)	Resultado incoerente produzido por meio de uma ação no sistema;
FALHA (*FAILURE*)	Incapacidade do *software* exercer a função para a qual foi desenvolvido.

Isolar um defeito

Isolar um defeito consiste em determinar sob quais condições ele ocorre. O objetivo é encontrar as causas dentro de um programa que estão ocasionando falhas e isso implica descobrir em qual linha de código ocorre uma falha como um crash (ou seja, o programa é abortado).

Estabilizar um programa

Estabilizar um programa é o termo geralmente utilizado para referir-se a correções que resultam na diminuição na frequência de falhas. Um programa estável apresenta poucas falhas – um indicativo de que deve possuir poucos defeitos. De maneira bastante geral, a estabilidade está ligada à idade de um programa.

Qualidade e bugs II: catástrofes

Os defeitos de software que levam ao crash do programa são certamente bastante inconvenientes. Mas, como foi dito antes, não constituem o único aspecto que determina a qualidade de um produto: há outros fatores, como o preço, que não devem ser desprezados quando se busca determinar a qualidade.

Qualidade e bugs II: catástrofes

Erros de software já foram responsáveis por prejuízos milionários e mesmo a perda de vidas humanas. A importância de garantir a qualidade é evidente à luz desta citação de Pressman [2002]: "O software de computadores... está embutido em sistemas de todas as naturezas: de transportes, médicos, de telecomunicações, militares, de processos industriais, de produtos de escritório,... a lista é quase sem-fim".

Ariane 501

Em 4 de junho de 1996, foi lançado o primeiro foguete Ariane 5. Decorridos 40 segundos da seqüência de lançamento e a uma altitude de 3.700 metros, o foguete se desviou de sua trajetória e se autodestruiu com uma explosão.

Therac-25

O Therac-25 era uma máquina utilizada em terapia radiológica. Diferente de suas versões anteriores, era totalmente controlado por um computador, um PDP-11.

Qualidade e o SWEBOK

Uma das áreas de computação – a Engenharia de Software – passou por um estudo de uma comissão internacional de especialistas, visando a uma definição das fronteiras que a delimitam. Esse estudo foi conduzido no âmbito da IEEE e chama-se SWEBOK (Software Engineering Body Of Knowledge, ou Corpo de Conhecimento de Engenharia de Software) [SWEBOK, 2004]

Qualidade e o SWEBOK

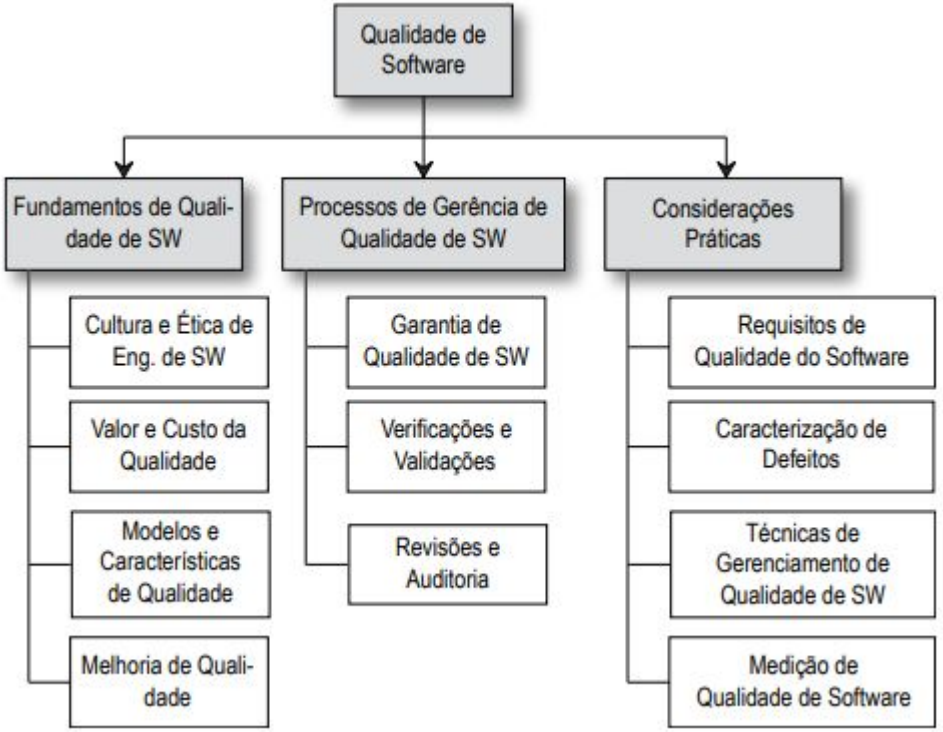
A Engenharia de Software é dividida no SWEBOK em um total de onze áreas de conhecimento (KA: Knowledge Area): requisitos, gerência de engenharia, projeto, métodos e ferramentas de engenharia, construção, processo de engenharia, testes, qualidade, manutenção, disciplinas relacionadas e gerência de configuração.

Qualidade e o SWEBOK

Em relação à qualidade, o SWEBOK fez uma distinção entre técnicas estáticas e dinâmicas. As primeiras aparecem sob a área de conhecimento Qualidade, enquanto as últimas figuram na área de Testes. A norma internacional ISO/IEC 25000 SQuaRE, que trata da qualidade de produtos de software, abrange esses dois tópicos.

Qualidade e o SWEBOK

Cada área de conhecimento no SWEBOK é subdividida em até dois níveis, formando uma estrutura hierárquica para catalogar os assuntos. No caso da área de qualidade, essa organização hierárquica é:



Qualidade e o SWEBOK

Fundamentos de qualidade: Este tópico abrange sobretudo a noção de qualidade, ou seja, sua definição. Essa definição, no caso de um produto, materializa-se por meio da definição de requisitos e estes dependem de um modelo. Um dos modelos mais importantes existentes atualmente é a norma SQuaRE, ISO/IEC 25000.

Qualidade e o SWEBOK

Processos de gerência de qualidade: Os processos de gerência abrangem todos os aspectos de construção do produto. Por conta disso, todos elementos de um projeto estão envolvidos: ferramentas como sistemas para controle de versão e linguagens, metodologias para revisão do produto, técnicas organizacionais e de administração de pessoas etc.

Qualidade e o SWEBOK

Considerações práticas: Este tópico contém observações de ordem prática, isto é, recomendações gerais sobre como transcorre a execução das atividades relacionadas com qualidade. Não há uma descrição explícita para este tópico no SWEBOK [2004].

Exercícios

Desenvolva os exercícios da lista 1.

Bibliografia

KOSCIANSKI, André; SOARES, Michel dos Santos. Qualidade de software: aprenda as metodologias e técnicas mais modernas para o desenvolvimento de software. São Paulo: Novatec, 2ª ed., 2007.

PRESSMAN, Roger S. Engenharia de software. 5. ed. Rio de Janeiro: McGraw Hill, 2002.

SOMMERVILLE, Ian. Engenharia de Software. 8. ed. São Paulo: Addison Wesley, 2007