

Qualidade, Qualidade de Software e Garantia da Qualidade de Software São as Mesmas Coisas?

Fábio Martinho. obtido [on-line] na URL <http://www.testexpert.com.br/?q=node/669>, em 11/03/2008.

Segundo a NBR ISO 9000:2005, "qualidade é o grau no qual um conjunto de características inerentes satisfaz aos requisitos". Ou seja, pode-se afirmar que se algum produto ou serviço atende aos requisitos especificados, este mesmo produto ou serviço possui a qualidade desejada.

A qualidade pode ser medida através do grau de satisfação em que as pessoas avaliam determinado produto ou serviço. No entanto, esse produto ou serviço pode ter qualidade para algumas pessoas e para outras nem tanto, ou seja, a qualidade é algo subjetivo.

Conceituar desta forma então o termo qualidade se torna uma tarefa muito difícil, pois elementos intrínsecos estão enraizados no intelecto de cada ser.

O termo TQM (*Total Quality Management*), amplamente usado nas organizações, também descreve uma abordagem para a melhoria da qualidade. De acordo com Kan (2002), "O termo tem tomado vários significados, dependendo de quem interpreta e como se aplica." (KAN, 2002, p. 30). Independente dos seus vários tipos de implementação, os elementos chave do TQM podem ser resumidos conforme Figura 1, abaixo:

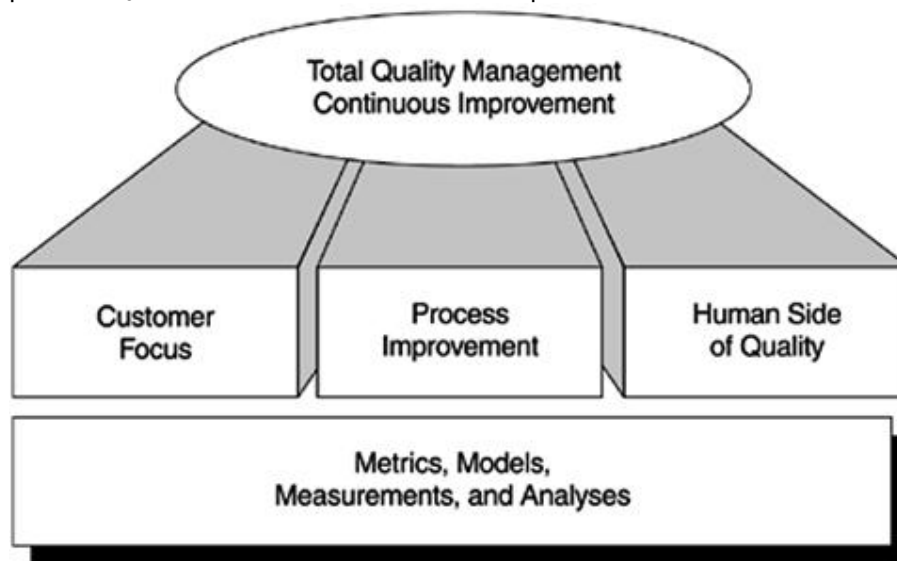


Figura 1 - Elementos chave do TQM.

a) **Foco do Cliente (*Customer Focus*)** - O objetivo é atingir a satisfação total do cliente. O foco do cliente inclui o estudo das necessidades e vontades do cliente, coleta de requisitos do cliente e a medição e gerenciamento da satisfação do cliente.

b) **Melhoria de Processo (*Process Improvement*)** - O objetivo é reduzir as variações de processo e atingir a melhoria da qualidade contínua. Este elemento inclui ambos os processos de negócio e o processo de desenvolvimento do produto. Através da melhoria de processo, a qualidade do produto será reforçada.

c) **Lado Humano da Qualidade (*Human Side of Quality*)** - O objetivo é criar a cultura de qualidade por toda a empresa. As áreas de foco incluem liderança, apoio da alta gerência, participação total de todos os colaboradores da empresa, e outros fatores humanos como sociais e psicológicos.

d) **Métricas, Modelos, Medições e Análises (*Metrics, Models, Measurement and Analysis*)** - O objetivo é direcionar a melhoria contínua em todos os parâmetros da qualidade por um sistema de medição orientado a metas.

No contexto de sistemas de informação e softwares o conceito da ISO aplica-se na sua totalidade, sendo que os usuários finais sempre terão as expectativas de um alto padrão de qualidade para que suas tarefas sejam sempre executadas da maneira mais adequada possível.

Tian (2005) afirma que as expectativas de qualidade para sistemas de software estão relativamente ligadas em dois aspectos:

a) "O sistema de software deve fazer o que é suposto que se faça. Em outras palavras, eles devem fazer certo as coisas". (TIAN, 2005, p. 35)

b) "Eles devem realizar estas tarefas específicas corretamente e satisfatoriamente. Em outras palavras, eles devem fazer as coisas certas". (TIAN, 2005, p. 35)

Indiretamente, as duas afirmações de Tian (2005) estão com base na ISO 9000:2005 sendo que o sistema deve fazer o que se espera que ele faça, de acordo com seus requisitos levantados e especificados.

Contudo, a qualidade possui alguns princípios básicos, como:

- Tentar prevenir defeitos ao invés de consertá-los;
- Ter a certeza que os defeitos que foram encontrados, sejam corrigidos o mais rápido possível.
- Estabelecer e eliminar as causas, bem como os sintomas dos defeitos;
- Auditar o trabalho de acordo com padrões e procedimentos previamente estabelecidos.

Segundo ainda a NBR ISO 8402, o conceito de qualidade é "A totalidade das características de uma entidade que lhe confere a capacidade de satisfazer às necessidades explícitas e implícitas". As necessidades explícitas são aquelas expressas na definição formal de requisitos propostos pelo cliente. Esses requisitos definem as condições em que o produto ou serviço devem ser utilizados bem como seus objetivos, funções e o desempenho esperado. Já as necessidades implícitas são aquelas que, embora não expressas pelo cliente nos documentos de requisitos, são necessárias para o usuário. Estão incluídos nessas classes tanto os requisitos que não precisam ser declarados por serem óbvios como aqueles requisitos que não são percebidos como necessários no momento que o produto foi desenvolvido, mas que pela gravidade de suas consequências devem ser atendidos.

A qualidade, seja ela usada no contexto de software ou de produtos e serviços, hoje não mais é uma obrigação e um diferencial das empresas. A mesma se tornou um padrão em qualquer ramo de atividade e indústria sendo assim necessária para garantir a satisfação do cliente. Qualidade hoje em dia, não é apenas um diferencial de mercado para as empresas conseguirem vender e lucrar mais, é um pré-requisito que se deve conquistar para conseguir colocar o produto ou serviço no Mercado Global. De acordo com Jack Welch, "A qualidade é a nossa melhor garantia da fidelidade do cliente, a nossa mais forte defesa contra a competição estrangeira e o único caminho para o crescimento e para os lucros."

Qualidade de Software

Diante dessa complexidade na definição da palavra qualidade, Pressman (2005) sugere que a qualidade de software seja implementada e não somente uma idéia ou desejo que uma organização venha a ter. Para tanto, Pressman (2005) faz as seguintes colocações sobre qualidade de software:

- a) "Definir explicitamente o termo qualidade de software, quando o mesmo é dito";(PRESSMAN, 2005, p. 193)
- b) "Criar um conjunto de atividades que irão ajudar a garantir que cada produto de trabalho da engenharia de software exiba alta qualidade"; (PRESSMAN, 2005, p. 193)
- c) "Realizar atividades de segurança da qualidade em cada projeto de software";(PRESSMAN, 2005, p. 193)
- d) "Usar métricas para desenvolver estratégias para a melhoria de processo de software e, como consequência, a qualidade no produto final"; (PRESSMAN, 2005, p. 193)

Sendo assim, a busca constante pela qualidade não se faz apenas no começo do projeto ou no seu final realizando testes, mas sim e um processo que visa abranger toda a engenharia de software bem como a colaboração de todos os membros do time do projeto.

Uma possível definição mais abrangente e completa para qualidade de software seria a proposta por Bartié (2002): "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)

Alguns modelos de qualidade de software também são citados por Pressman (2005). Há o que McCall e Cavano (1978) sugerem como métricas para qualidade de software. Conhecido como Fatores da Qualidade, estes fatores avaliam o software em três pontos distintos: Transição do Produto, Revisão do Produto e Operação do Produto. Na Figura 2 são mostrados os Fatores da Qualidade de McCall:



Assim como o modelo proposto por McCall e Cavano (1978), "a Hewlett-Packard desenvolveu também um modelo que referencia fatores da qualidade de software e que primeiramente publicado por Grady and Caswell (1987), denominado FURPS: *Functionality, Usability, Reliability, Performance e Supportability*. Estes fatores estabelecem as métricas de qualidade de software para cada fase do processo de engenharia de software". (PRESSMAN, 2005, p. 539)

Além desse modelos de métricas para qualidade de software, nota-se que a constante busca pela mesma se tornou uma atividade essencial dentro das empresas. Colocando-se todos esses conceitos dentro do contexto apresentado, podemos dizer que "qualidade não é uma fase do ciclo de desenvolvimento de software ... é parte de todas as fases". (BARTIÉ, 2002, p. 16)

Portanto, é necessário um planejamento adequado para que a qualidade de software seja atingida, conforme a definição de qualidade que deverá ser alcançada. Para isso são necessários modelos, padrões, procedimentos e técnicas para atingir essas metas de qualidade propostas. Para tanto, todas as etapas do ciclo de vida de engenharia de software devem ser contempladas com atividades que visam garantir a qualidade tanto do processo quanto do produto.

Software Quality Assurance

Segundo Lewis (2004), uma definição formal de *Software Quality Assurance*(SQA) é "atividades sistemáticas fornecendo evidências para o uso pretendido para o produto total de software". (LEWIS, 2004, p. 18)

Sendo assim, podemos ainda definir como *Quality Assurance* "o conjunto de atividades de apoio para fornecer confiança de que os processos estão estabelecidos e estão continuamente melhorados para produzir produtos que atendam as especificações e que sejam adequados para o uso pretendido". (LEWIS, 2004, p. 18)

Com isso, o SQA envolve todo o processo de desenvolvimento de software fazendo as devidas monitorações e melhorias de processos pertinentes, fazendo com que os padrões, procedimentos acordados estão sendo seguidos e garantindo que problemas são encontrados e ações corretivas são tomadas. Esse tipo de ação é orientada a prevenção. O IEEE 610.12-1990 cita qualidade de software como "(1) Um padrão planejado e sistemático de todas as ações necessárias para fornecer confiança adequada que um item ou produto está em conformidade com os requisitos técnicos estabelecidos. (2) Um conjunto de atividades projetadas para avaliar o processo pelo qual produtos são desenvolvidos ou manufaturados".

O SQA é também entendida e formada por um grupo de pessoas relacionadas e empregadas através de todo o ciclo de vida de engenharia de software que positivamente influenciam e quantificam a qualidade do software que está sendo entregue. Como já foi dito, o SQA não é somente uma atividade associada exclusivamente com atividades de desenvolvimento de software, mas sim atividades que se expandem durante todo o ciclo de vida de desenvolvimento de software.

Portanto, isso consiste em realizar a qualidade tanto do processo quanto to produto. No processo, podemos quantificar a sua qualidade através de métricas para qualidade de software e no produto com as técnicas de verificação e validação. Essas atividades podem ser por exemplo avaliações como as citadas pela ISO 9000, auditorias, inspeções formais, teste de software, revisões. Ainda no processo podemos usar os métodos de garantia da qualidade no formato de auditorias e reportes para a alta gerência, além de avaliações constantes do processo e análise estatística de controle do processo. No produto os métodos de garantia da qualidade são

revisões, inspeção formal e teste de software, além de revisão dos resultados do teste de software realizada por *experts*, auditorias do produto e testes realizados pelo cliente.

No entanto, empresas que não possuem o grupo ou processos de SQA tendem a mostrar os seguintes indicadores de falta de qualidade, conforme Lewis (2004):

- a) O software que foi entregue freqüentemente apresenta falhas;
- b) Inaceitáveis conseqüências de falhas de sistemas, desde financeiras até cenários reais de aplicação;
- c) Sistemas não estão freqüentemente disponíveis para uso pretendido;
- d) Sistemas são freqüentemente muito caros;
- e) Custo de detectar e remover defeitos são excessivos.

Por outro lado, empresas que possuem o grupo ou processo de SQA implementados e a sua aplicação de maneira adequada e correta mostra que:

- a) A remoção de erros acontece no momento em que se é barato corrigir;
- b) Melhoria da qualidade do produto;
- c) O SQA é um recurso para a melhoria de processo;
- d) Estabelecimento de um banco de dados de métricas como: planejamento, taxas de falhas e outros indicadores da qualidade;

Lewis (2004) cita as atividades mais comuns do SQA, sendo estas categorizadas como: Teste de Software (Verificação e Validação), Gerenciamento de Configuração de Software e Controle da Qualidade. Na Figura 3, podemos ver a relação entre essas três principais atividades juntamente com Padrões, Procedimentos, Convenções e Especificações:

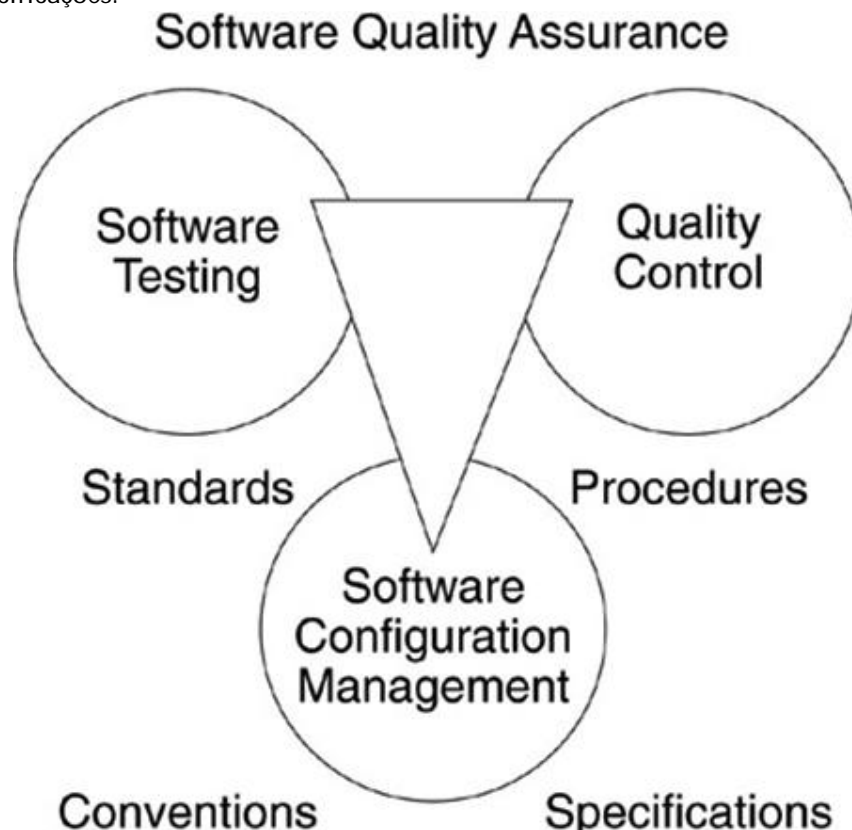


Figura 3 - Componentes do SQA.

a) **Teste de Software (Software Testing):** Conforme Lewis (2004), "É uma estratégia popular para o gerenciamento de risco". (LEWIS, 2004, p. 19)

O teste de software é usado para verificar que requisitos funcionais e não-funcionais foram devidamente implementados. A limitação dessa abordagem (Teste de Software), entretanto, é que na fase em que ele acontece, muitas vezes é difícil de se conseguir alguma qualidade no produto. Isso porque muitas empresas ainda usam o modelo *Waterfall*, ou modelo cascata, que foca justamente a atividade de teste de software somente no final do modelo, ou seja, caso algum defeito seja encontrado (erros em requisitos, por exemplo) todo ciclo deverá ser inicializado novamente. O teste de software na verdade, foca quase que exclusivamente as atividades de verificação e validação.

b) **Controle da Qualidade (Quality Control):** O controle da qualidade é definido como um processo e métodos usados para monitorar o trabalho e observar se os requisitos estão sendo satisfeitos. O foco é

justamente em revisões e remoção de defeitos antes mesmo do envio dos produtos. O controle da qualidade deve ser de responsabilidade da unidade organizacional que produz o produto. No entanto, é possível ter o mesmo grupo que constrói o produto, que realize também o controle da qualidade, ou estabelecer um grupo de controle da qualidade separado ou departamento dentro da mesma unidade organizacional que desenvolve o produto.

O controle da qualidade consistem de *checklists* bem definidos em um produto que é especificado no plano de garantia da qualidade. Um exemplos clássico de controle da qualidade são as inspeções de software. A inspeção é o grau mais maduro e formal dentro das revisões, sendo necessária uma preparação prévia, participantes definidos adequadamente e critérios de entrada e saída bem definidos.

O controle da qualidade é projetado para detectar defeitos e corrigir esses defeitos encontrados, enquanto que a garantia da qualidade é orientada através da prevenção de defeitos.

c) **Gerenciamento de Configuração de Software (SCM - Software Configuration Management):** O SCM é responsável por identificar, rastrear e controlar mudanças nos elementos do software de um sistema. O SCM controla a evolução do sistema de software , gerenciando versões dos componentes de software e seus relacionamentos. O propósito é identificar todos os componentes inter-relacionados do software e para controlar sua evolução através das várias fases no ciclo de vida de desenvolvimento de software.

SCM é uma disciplina que pode ser aplicada para atividades incluindo: desenvolvimento de software, controle de documentação, problemas de rastreamento, controle de mudanças e manutenção. O SCM ainda consiste de atividades que asseguram que arquitetura e codificação são definidas e não podem ser mudados sem uma revisão dos efeitos da mudança e sua documentação. Isso porque conforme definição do SCM é controlar o código-fonte e a sua documentação associada fazendo com que o código-fonte final e suas descrições estão consistentes e representam os itens que estavam revisados e testados.

Par que ainda estes três principais componentes funcionem corretamente, o sucesso do programa de garantia da qualidade de software também depende de uma coerente coleção de padrões, procedimentos, convenções e especificações, conforme Figura 3.

A combinação de todos esses componentes e suas melhores práticas é o que chamamos de *Software Quality Assurance*, e que por sua vez todo esse trabalho é realizado por pessoas, garantindo então a qualidade de software do produto final entregue ao cliente ou usuário final.

Conforme Figura 3 Componentes do SQA, o foco será exclusivamente o componente Teste de Software.

Para que também toda essa estrutura esta devidamente documentada e aceita por todos os membros do time do projeto, é necessário um planejamento adequado. Esse planejamento é feito no *Software Quality Assurance Plan* ou Plano de Garantia da Qualidade de software. Segundo Lewis (2004), "O plano de garantia da qualidade de software é um resumo ou esboço das medidas de qualidade para garantir níveis de qualidade dentro do esforço do desenvolvimento de software".(LEWIS, 2004, p. 22)

O plano é usado como um *baseline* para comparar os níveis atuais de qualidade durante o desenvolvimento com os níveis planejados de qualidade. "O plano de SQA provê o *framework* e guias para o desenvolvimento de um código entendível e que seja de fácil manutenção"(LEWIS, 2004, p. 22).

Controle da Qualidade de Software versus Garantia da Qualidade de Software

Um dos grandes erros geralmente cometidos por pessoas e empresas é confundir os conceitos e aplicação dos termos Controle da Qualidade(*Quality Controle*) e Garantia da Qualidade(*Quality Assurance*). Embora usados de maneira errônea em muitos lugares, ambos os termos têm propósitos totalmente diferentes.

A **Tabela 1** - Diferenças entre Garantia da Qualidade e Controle da Qualidade mostra a diferença entre estas duas atividades:

Quality Assurance	Quality Control
1. Garantia da qualidade garante que o processo é definido e apropriado.	1. As atividades de controle da qualidade focam na descoberta de defeitos em i específicos.
2. Metodologia e padrões de desenvolvimento são exemplos de garantia da qualidade.	2. Um exemplo de controle da qualidade poderia ser: "Os requisitos definidos são os requisitos certos?".
3. Garantia da qualidade é orientada a processo.	3. Controle da qualidade é orientado a produto.
4. Garantia da qualidade é orientada a prevenção.	4. Controle da qualidade é orientado a detecção.
5. Foco em monitoração e melhoria de processo.	5. Inspeções e garantia de que o produto de

6. As atividades são focadas no início das fases no ciclo de vida de desenvolvimento de software.	trabalho atenda aos requisitos especificados.
7. Garantia da qualidade garante que você está fazendo certo as coisas e da maneira correta.	6. As atividades são focadas no final das fases no ciclo de vida de desenvolvimento de software.
	7. Controle da qualidade garante que os resultados do seu trabalho são os esperados conforme requisitos.

Com isso, pode-se afirmar que o teste de software é uma das atividades de controle da qualidade, ou seja, o teste de software é orientado a produto e está dentro do domínio do controle da qualidade.

Qualidade de Software segundo o SWEBOK

O SWEBOK (*Software Engineering Body of Knowledge*) versão 2004 do IEEE (*Institute of Electrical and Electronics Engineers*) apresenta em uma das suas KA's (*Knowledge Areas*) a KA de *Software Quality*. Abaixo na Figura 4, pode-se ver a estrutura da KA's de *Software Quality*:

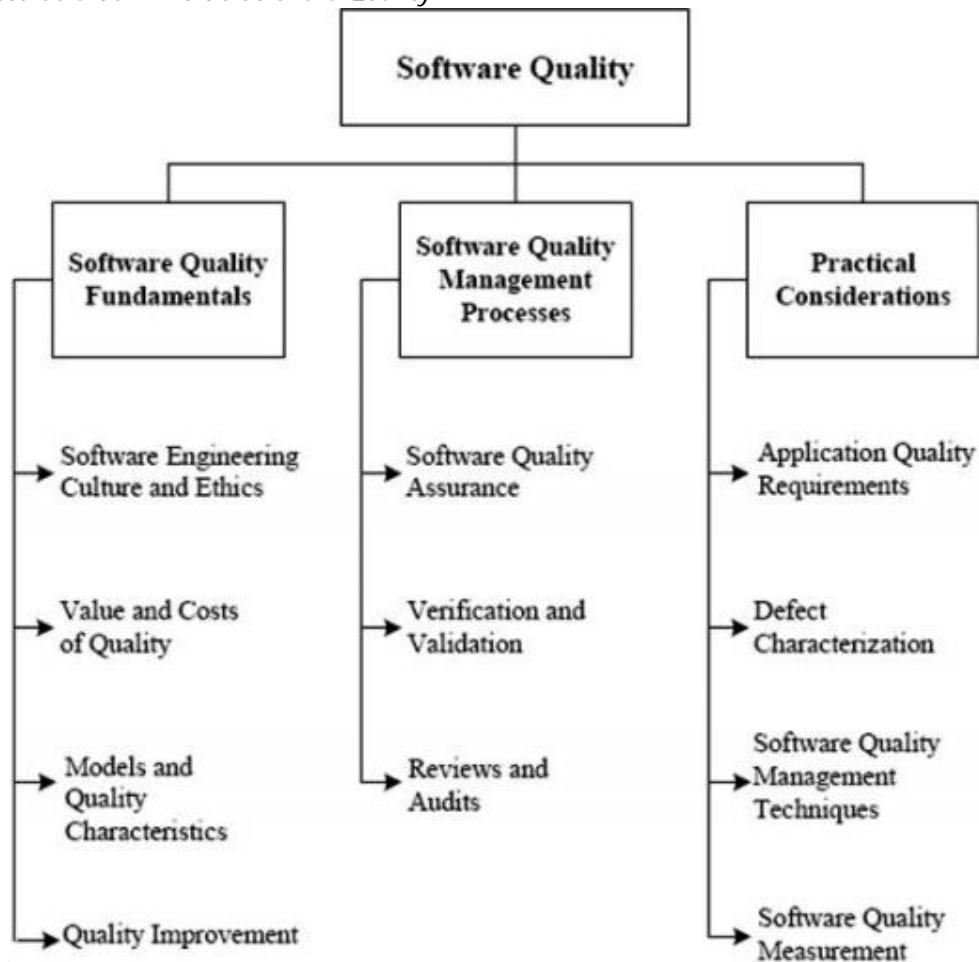


Figura 4 - SWEBOK - KA de *Software Quality*.

O SWEBOK descreve ainda inúmeras maneiras para se atingir a qualidade sendo que esta KA em específico cobre as técnicas estáticas para detecção de defeitos, não sendo requerido que o software esteja sendo executado, enquanto que as técnicas dinâmicas são cobertas pela KA de *Software Testing*.

Custo da Qualidade de Software

A qualidade não é totalmente algo que podemos obter de forma gratuita. Para tanto, investimentos financeiros, treinamentos, softwares e outras iniciativas precisam ser realizadas em adição para a busca da qualidade de software como um todo.

Segundo Bartié (2002), "Um dos maiores desafios a ser considerados é estabelecer um modelo de custos relacionados a implantação de um processo de garantia da qualidade de software." (BARTIÉ, 2002, p. 29)

A Figura 5 mostra um modelo de custo de qualidade de software proposto por Bartié (2002):

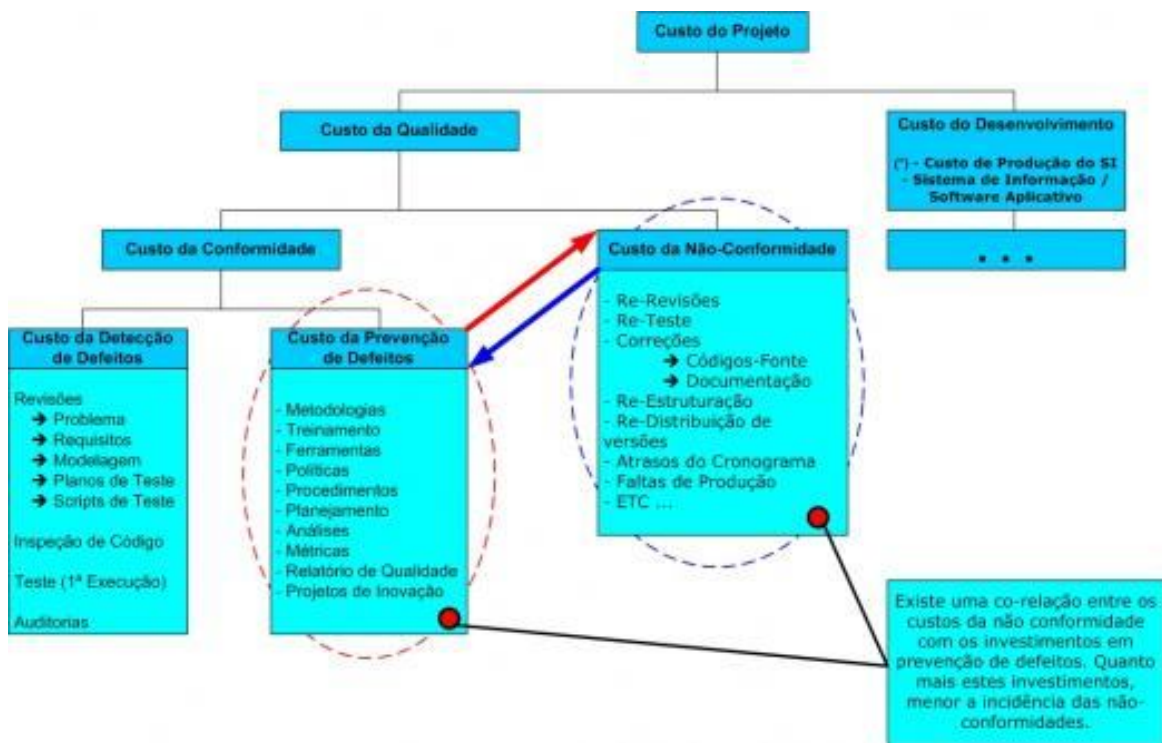


Figura 5 - Modelo de Custo de Qualidade de Software.

No modelo apresentado, Bartié (2002) propõe a criação de duas categorias separadas para os custos relacionados a conformidade e o custo da não-conformidade:

a) **Custo da Detecção de Defeitos:** Pode-se aqui fazer claramente referências para o termo controle da qualidade, ou seja, o foco está exatamente no produto. As atividades aqui realizadas são orientadas ao produto desenvolvido, e estas incluem:

- Revisões de requisitos;
- Revisões de Modelagem;
- Revisões de Planos de Teste;
- Inspeções de código;
- Testes de Software.

b) **Custo da Prevenção de Defeitos:** Assim como detecção de defeitos está associada ao controle da qualidade, a prevenção de defeitos está associada a garantia da qualidade, ou seja, o foco está exatamente no processo. As atividades aqui realizadas são orientadas ao processo, e estas incluem:

- Definição de Metodologias;
- Treinamentos;
- Ferramentas de apoio ao processo de desenvolvimento;
- Definição de Políticas;
- Procedimentos;
- Padrões;
- Especificações e convenções;
- Planejamento do SQA;
- Relatórios de Qualidade para melhoria de processo.

c) **Custo da Não-Conformidade:** Por outro lado, o custo da não-conformidade está relacionado às perdas que o projeto terá, não optando pela detecção e prevenção de defeitos:

- Re-revisões;
- Re-testes;
- Correções de código-fonte e documentação muito constantes;
- Reestruturação;
- Redistribuição das versões do software;
- Atrasos no cronograma;
- Falhas na produção.

Com isso, "O modelo apresentado deverá ser associado a todas as atividades de um processo de engenharia de software. Em todos os projetos a serem construídos ou modificados, todas as atividades deveriam ter uma política de alocação de custos semelhante ao modelo apresentado." (BARTIÉ, 2002, p. 29)

Qualidade de Software segundo a ISO 9126-1

A ISO também apresenta as características da qualidade de software através da norma NBR ISO/IEC 9126-1. A antiga norma brasileira para as características da qualidade de software era a NBR 13.596. No entanto, a mesma se integrou a ISO, formando a NBR ISO/IEC 9126-1. Na Figura 6, são mostradas essas características juntamente com suas subcaracterísticas:

Características	Subcaracterísticas
Funcionalidade (satisfação das necessidades)	<ul style="list-style-type: none">• Adequação (execução do que é apropriado)• Acurácia (execução de forma correta)• Interoperabilidade (interação com quem deve)• Conformidade (aderência às normas)• Segurança de acesso (bloqueio de uso não autorizado)
Confiabilidade (imunidade a falhas)	<ul style="list-style-type: none">• Maturidade (frequência das falhas)• Tolerância a falhas (forma de reação à falhas)• Recuperabilidade (forma de recuperação de falhas)
Usabilidade (facilidade de uso)	<ul style="list-style-type: none">• Intelegibilidade (facilidade de entendimento)• Apreensibilidade (facilidade de aprendizado)• Operacionalidade (facilidade de operação)
Eficiência (rápido e "enxuto")	<ul style="list-style-type: none">• Tempo (tempo de resposta, velocidade de execução)• Recursos (recursos utilizados)
Manutenibilidade (facilidade de manutenção)	<ul style="list-style-type: none">• Analisabilidade (facilidade de encontrar falha)• Modificabilidade (facilidade de modificar)• Estabilidade (baixo risco quando de alterações)• Testabilidade (facilidade de testar)
Portabilidade (uso em outros ambientes)	<ul style="list-style-type: none">• Adaptabilidade (facilidade de se adaptar a outros ambientes)• Capacidade para ser instalado (facilidade de instalar em outros ambientes)• Conformidade (aderência a padrões de portabilidade)• Capacidade para substituir (facilidade de ser substituído por outro)

Figura 6 - NBR ISO/IEC 9126-1 - Características da Qualidade de Software.

- Funcionalidade(Satisfação das Necessidades):** É a capacidade do produto de software de prover funcionalidades que satisfaçam as necessidades quando o software está em uso dentro das condições especificadas.
- Confiabilidade(Imunidade a Falhas):** É a capacidade do produto de software de manter um nível especificado de performance quando o software está em uso dentro das condições especificadas.
- Usabilidade(Facilidade de Uso):** É a capacidade do produto de software de ser entendido, aprendido, usado e atrativo quando o software está em uso dentro das condições especificadas.
- Eficiência(Rápido e "Enxuto"):** É a capacidade do produto de software de prover performance apropriada, relativa ao conjunto de recursos usados quando o software está em uso dentro das condições especificadas.
- Manutenibilidade(Facilidade de Manutenção):** É a capacidade do produto de software de ser mudado. Modificações incluem correções, melhorias ou adaptações do software de mudar em um ambiente, e em requisitos e especificações funcionais.
- Portabilidade(Uso em outros Ambientes):** É a capacidade do produto de software de ser transferido de um ambiente para outro.

Além da NBR ISO/IEC 9126-1, existem ainda outras normas da série 9126, as quais são:

- ISO/IEC 9126-2** - Métricas Externas: Podem ser aplicadas para um produto não executável durante os estágios de desenvolvimento. Medem a qualidade de produtos intermediários e predizem a qualidade do produto final.
- ISO/IEC 9126-3** - Métricas Internas: Utilizadas para medir a qualidade do software através do comportamento do sistema ou de parte dele. Só podem ser usadas durante a fase de testes do ciclo de vida e durante a operação do sistema.
- ISO/IEC 9126-4** - Métricas da Qualidade do Uso: medem se o produto atende ou não as necessidades dos usuários, fazendo-os atingir seus objetivos com efetividade, produtividade, segurança e satisfação. Só podem ser usadas no ambiente real ou em uma aproximação do ambiente real.

A Qualidade segundo o PMBOK do PMI

O PMBOK (*Project Management Body Of knowledge*) do PMI (*Project Management Institute*), na sua versão 2004 apresenta o gerenciamento da qualidade do projeto, conforme Figura 7 abaixo:

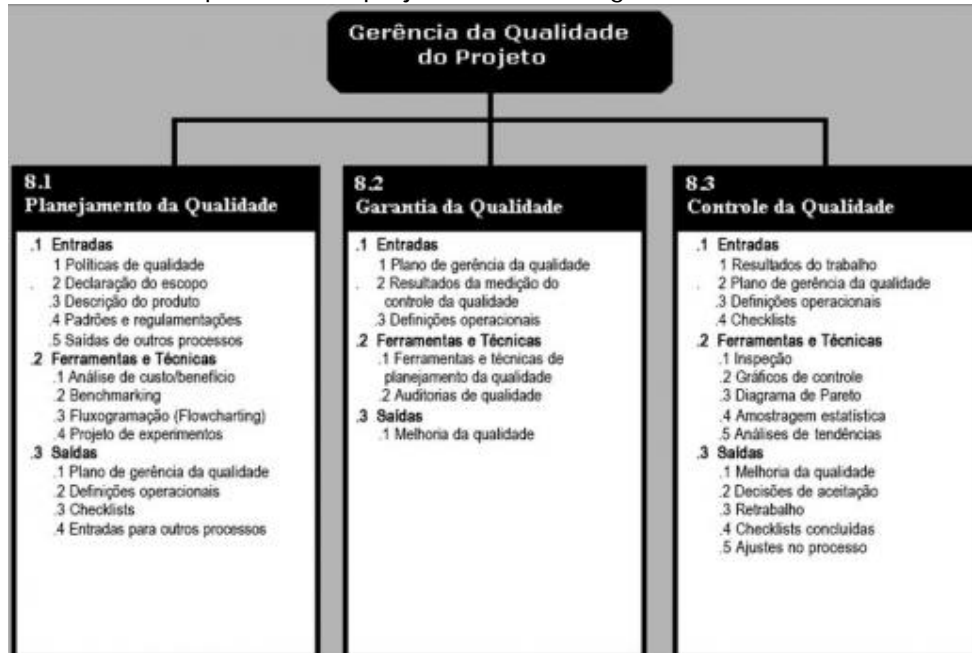


Figura 7 - Gerenciamento da Qualidade do Projeto segundo o PMBOK do PMI.

De acordo com o PMBOK (2004), "Os processos de gerenciamento da qualidade do projeto incluem todas as atividades da organização executora que determinam as responsabilidades, os objetivos e as políticas de qualidade, de modo que o projeto atenda às necessidades que motivaram sua realização. Eles implementam o sistema de gerenciamento da qualidade através da política, dos procedimentos e dos processos de planejamento da qualidade, garantia da qualidade e controle da qualidade, com atividades de melhoria contínua dos processos conduzidas do início ao fim, conforme adequado". (PMI, 2004, p.179)

Com isso os três principais processos são:

- Planejamento da Qualidade:** "Identificação dos padrões de qualidade relevantes para o projeto e determinação de como satisfazê-los." (PMI, 2004, p. 179)
- Garantia da Qualidade:** "Aplicação das atividades de qualidade planejadas e sistemáticas para garantir que o projeto emprega todos os processos necessários para atender aos requisitos." (PMI, 2004, p. 179)
- Controle da Qualidade:** "Monitoramento de resultados específicos do projeto a fim de determinar se eles estão de acordo com os padrões relevantes de qualidade e identificação de maneiras de eliminar as causas de um desempenho insatisfatório." (PMI, 2004, p. 179)

Como se pode notar é evidente a semelhança entre os conceitos usados no PMBOK e os conceitos da própria ISO. Com isso, é possível ainda relacionar estes três processos do PMBOK com as definições de controle da qualidade e garantia da qualidade de software apresentados anteriormente.

[Fábio Martinho Campos](#), CBTS® CST®

CBTS - Certified Brazilian Tester in Software

CST - Certified Software Testing

Especialista em Qualidade de Software e Testes de SI/Softwares

Software Quality and SI/Softwares Test Specialist