QUALIDADE DE SOFTWARE

SOLUÇÕES EDUCACIONAIS INTEGRADAS

Métricas de qualidade de *software*

Whesslley Rimar Bezerra

OBJETIVOS DE APRENDIZAGEM

- Explicar a medição de qualidade de *software* e suas finalidades.
- Descrever o processo de medição de produto e suas fases.
- Reconhecer os indicadores de qualidade de *software*.

Introdução

Medir a qualidade das coisas faz parte do universo humano. Sempre procuramos encontrar um nível de equilíbrio que defina um produto ou serviço como de alta, média ou baixa qualidade. A qualidade nos faz querer adquirir ou abrir mão da compra de um produto ou serviço, sendo um fator preponderante na tomada de decisões. Na área de desenvolvimento de sistemas, não é diferente: a qualidade é aferida por métricas que nos fazem refletir sobre o desempenho de um *software*, auxiliando os gestores a tomarem decisões embasados nos resultados dessas métricas.

Neste capítulo, você vai compreender as métricas de qualidade de *software*, conhecer o processo de medição de produto e, por fim, vai reconhecer os indicadores de qualidade de *software*.

Conceito

Antes de aprofundarmos a questão de medição da qualidade de um *software*, é preciso entender o que, de fato, significa o termo **qualidade**. Em uma definição geral, Silva (2017) afirma que qualidade é o nível mínimo esperado de utilidade de algum objeto, produto, serviço, etc. A qualidade está intimamente ligada às percepções que cada pessoa em particular tem com relação ao que é tangível (objetos, produtos, etc.) e intangível (serviços, funcionalidades de um produto, etc.).

De maneira análoga, a qualidade de *software* pode ser definida como o nível ou grau em que cada *software*, componente ou processo atende aos requisitos e às expectativas e necessidades do usuário (SILVA, 2017). A medição da qualidade de *software* tem duas finalidades principais (FIGUEIREDO, 2016):

- realizar previsões gerais sobre o sistema;
- identificar componentes anômalos.

A primeira se refere às possíveis modificações que o sistema pode vir a ter no futuro, quando são adicionadas novas funcionalidades, ou seja, se o sistema está apto a receber essas modificações. A segunda se refere à identificação de componentes que estejam passando por problemas que vão desde a implementação incorreta de determinada funcionalidade até os problemas de desempenho. É fundamental compreender que, para determinar a qualidade de um *software*, é preciso eleger os atributos que devem ser medidos e em quais circunstâncias eles devem passar pelo processo de medição.

Boehm, Brown e Lipow (1976) descrevem uma sequência de características (atributos) de qualidade de *software* (em formato de uma árvore), que foi traduzida e adaptada por Bueno e Campelo ([201-?]) e demonstrada na Figura 1, em que pode-se observar uma sequência lógica que representa o que é esperado de um *software*, em termos de qualidade. Como exemplo, seguindo as setas direcionais da árvore, percebe-se que um *software* de usabilidade bem definida é, consequentemente, confiável, eficiente e projetado com base em conceitos da engenharia humana. Além deste exemplo, a Figura 1 mostra muitos outros atributos que devem ser considerados na medição da qualidade de um *software*, como portabilidade, facilidade de teste, facilidade de entendimento e facilidade de modificação.

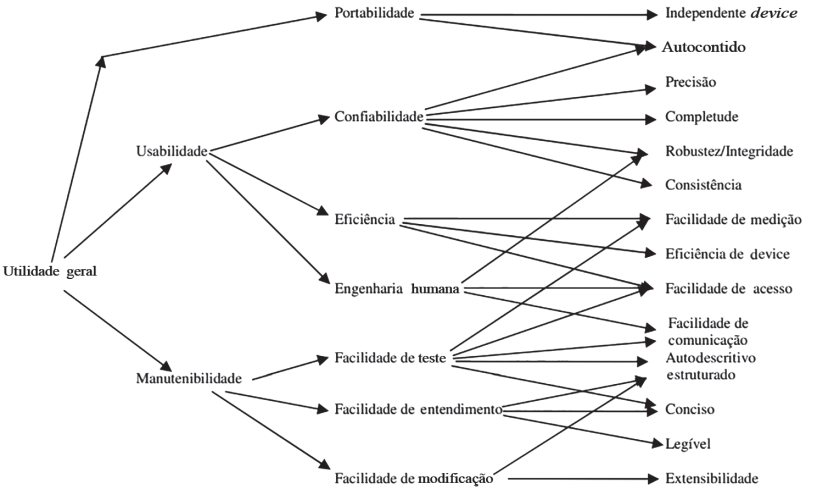

Figura 1. Atributos de qualidade de *software*.
Fonte: Bueno e Campelo ([201-?], documento *on-line*).

Ao falarmos em qualidade de *software*, é preciso compreender também os atributos internos e externos de qualidade. Veja a seguir (SANT'ANNA, 2004).

- Os **atributos internos** de um *software* são características que podem ser medidas ao se analisar o *software* diretamente, não sendo necessário executar o código para tal. Esses atributos são identificados facilmente ao analisar os componentes do *software*. Alguns exemplos de atributos internos são o tamanho do *software* (linhas de código), a complexidade, a dependência entre componentes, etc.
- Os **atributos externos** são aqueles que só podem ser analisados quando o *software* está em execução. Neste caso, o comportamento do *software* é analisado minuciosamente, não sendo verificado apenas o relacionamento entre os componentes.

Saiba mais

A engenharia humana é um campo da engenharia que estuda os impactos humanos dos projetos desta área. Esse campo pode envolver questões de ergonomia, segurança do trabalho, etc.

Medição de produto

Ao medir um *software*, deve-se perceber que o foco da avaliação é o produto de *software* em si. Como vantagens, produzir *software* de qualidade gera benefícios que colocam o *software*, o processo e a empresa em posições de destaque. Duas dessas vantagens são descritas a seguir.

- **Competitividade:** um *software* bem produzido faz com que a empresa se mantenha assertiva e competitiva no mercado, vendendo-o ou licenciando-o em larga escala.
- **Qualidade no processo:** um *software* de qualidade é reflexo de um processo de desenvolvimento bem definido e estruturado, ou seja, quando um sistema é considerado de ótima qualidade significa que o processo que o levou a ser construído seguiu um fluxo organizado, com fases bem orquestradas por uma equipe em sincronia.

Contudo, uma desvantagem em relação às métricas de *software*, sejam elas relacionadas à medição de qualidade ou não, é que não há uma padronização para elas, ou seja, é provável que a forma de medir a qualidade de *software* varie de um *software* para outro ou de um contexto para outro. Além disso, elas não são aplicáveis a todos os tipos de *software*, por isso, em muitos casos, outros tipos de medições devem ser realizados para que sejam obtidos resultados mais realistas.

Um *software* é um produto e, como tal, deve ser submetido ao processo de medição comum a qualquer produto. Nele, podem ser utilizados dados históricos de projetos anteriores. As atividades desse processo ocorrem em cinco passos, descritos a seguir (FIGUEIREDO, 2016).

1. **Escolher medições a serem realizadas:** para definir as medições que serão utilizadas, uma alternativa é o Goal–Question–Metric (GQM), que representa os objetivos, questões e métricas. Assim, os objetivos devem ser definidos, ou seja, devem ser determinadas quais melhorias a empresa almeja alcançar. A partir daí, questões são formuladas para atender aos objetivos da medição, e métricas são definidas para que essas questões sejam respondidas.
2. **Selecionar componentes a serem avaliados:** talvez não seja necessário medir todo o *software*. Desta forma, podem ser selecionados apenas os módulos (componentes) desejáveis. Os critérios para essa escolha podem ser os componentes críticos da aplicação.
3. **Medir características de componentes:** os componentes selecionados na etapa anterior são medidos qualitativamente por meio de ferramentas. Além disso, as medidas são vinculadas em atributos de qualidade.
4. **Identificar medições anômalas:** após as medições, são identificados os componentes que possam apresentar falhas, erros ou quaisquer outros problemas que comprometam a qualidade do sistema em análise.
5. **Analisar componentes anômalos:** os componentes anômalos identificados são analisados minuciosamente, a fim de serem encontradas as suas inconsistências. Esses dados são consolidados em um relatório e encaminhados para ações posteriores.

Na Figura 2, pode-se observar como a sequência de fases é distribuída.

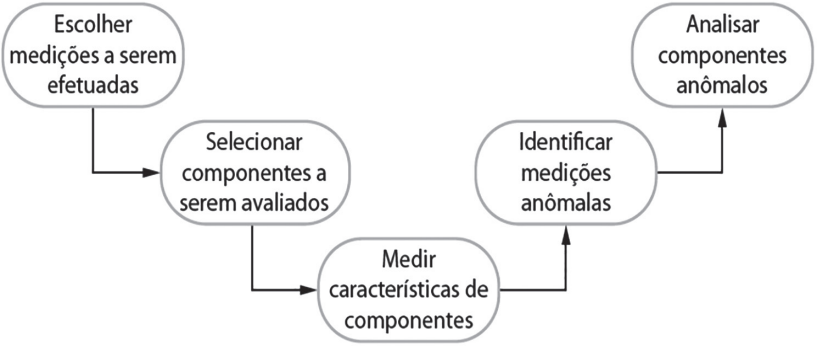

Figura 2. Fases da medição de produto.
Fonte: Figueiredo (2016, documento *on-line*).

Essas medições são classificadas, ainda, em duas categorias (RAMOS, 2005), descritas a seguir.

1. **Dinâmicas:** são iniciadas em *softwares*/sistemas que estejam em execução. Elas medem informações do *software*, como tempo de inicialização, o tempo para execução de determinada tarefa, a *performance* na tratativa de grandes volumes de dados, etc.
2. **Estáticas:** são coletadas em representações do *software*, ou seja, são medições realizadas com base no código-fonte do sistema, na organização do projeto, na documentação. Portanto, não são executadas no *software* em funcionamento.

Um dos parâmetros determinantes para a qualidade de *software* é a verificação da utilização de padrões no desenvolvimento, como as metodologias de desenvolvimento ágil ou as boas práticas de cada linguagem. Isso significa que a probabilidade de um *software* que siga os padrões de desenvolvimento e as boas práticas seguidas pela comunidade ser de boa qualidade é muito maior em relação àqueles que não seguem padrões.

Bueno e Campelo ([201-?]) ressaltam que a qualidade de *software* tem significados diferentes entre as pessoas. Portanto, o que se deve fazer para que ela atinja um nível aceitável é parametrizar os aspectos de qualidade nos quais se está interessado e, então, decidir medi-los. Antes de determinar esse grau de qualidade, os desenvolvedores precisam entrar em acordo sobre quais atributos denotam a qualidade do produto de *software*. Uma vez decidido, é só começar o processo de medição, utilizando as métricas necessárias.

Indicadores de qualidade de *software*

Nesta seção, vamos conhecer os principais indicadores de qualidade de *software*, utilizados como parâmetros para definir se um *software* ou parte dele e algumas de suas funcionalidades estão com a qualidade mínima aceitável para serem colocados em produção para uso dos clientes. Esses indicadores são descritos pela NBR ISO/IEC 9126-1, que trata da qualidade de produto de *software* (ABNT, 2003).

Funcionalidade

Corresponde às ações que podem ser executadas com o *software*, ou seja, suas funções, e devem atender às necessidades e expectativas dos usuários quando o *software* estiver em execução. Esse indicador deve atender ainda a algumas características, as quais são descritas no Quadro 1.

Quadro 1. Atributos de funcionalidade

Atributos de funcionalidade	Descrição
Adequação	São funções que têm a capacidade de executar tarefas especificadas pelo usuário.
Acurácia	Representa a precisão necessária provida pelo *software* para atingir os resultados esperados.
Interoperabilidade	Representa a capacidade de o *software* ser executado ou interagir com sistemas diferentes.
Segurança de acesso	Representa a capacidade de o *software* proteger os dados de seus clientes, usuários, produtos, serviços e quaisquer outras informações que estejam sob seu domínio. Os dados só devem ser acessíveis a quem tiver autorização para lê-los e/ou modificá-los.
Conformidade relacionada à funcionalidade	Representa a capacidade de o *software* estar em conformidade com as legislações vigentes relacionadas à funcionalidade.

Fonte: Adaptado de ABNT (2003).

Confiabilidade

Este indicador representa a capacidade de o *software* manter o nível de desempenho esperado quando executado por seus usuários em condições previamente especificadas. Algumas características devem ser atendidas nesse indicador, e são descritas no Quadro 2.

Quadro 2. Atributos de confiabilidade

Atributos de confiabilidade	Descrição
Maturidade	Representa a capacidade de o *software* impedir que erros/falhas recorrentes aconteçam.
Tolerância a falhas	Representa a capacidade de o *software* manter o desempenho mínimo esperado e especificado, mesmo em momentos de falhas causadas por *bugs*.
Recuperabilidade	Representa a capacidade de o *software* recuperar o seu estado normal de desempenho após uma falha ocasionada por algum erro.
Conformidade relacionada à confiabilidade	Representa a capacidade de o *software* estar em conformidade com as legislações vigentes relacionadas à confiabilidade.

Fonte: Adaptado de ABNT (2003).

Usabilidade

Este indicador se refere à capacidade de o *software* ser operado e manuseado pelo usuário de maneira fluida. A ideia é que o *software* possa ser intuitivo, de fácil compreensão e atraente ao usuário. Veja no Quadro 3 algumas características que devem ser atendidas para este indicador.

Quadro 3. Atributos de usabilidade

Atributos de usabilidade	Descrição
Inteligibilidade	Representa a capacidade de o *software* prover ao usuário a possibilidade de compreender se o sistema é apropriado para atender às suas necessidades. Além disso, o *software* permite que o usuário consiga entender como ele deve ser utilizado na execução de suas tarefas.
Apreensibilidade	Representa a capacidade de o *software* fornecer ao usuário a possibilidade de aprender a utilizar a aplicação.

(Continua)

(Continuação)

Atributos de usabilidade	Descrição
Operacionalidade	Representa a capacidade de o *software* fornecer ao usuário recursos que possibilitam o controle e a operacionalização do sistema.
Atratividade	Representa a capacidade de o *software* ser visualmente atraente ao usuário, por meio de cores bem contrastadas e que se combinam. Além disso, com uma estrutura de leiaute bem definida, o usuário terá vontade e curiosidade em utilizar o sistema.
Conformidade relacionada à usabilidade	Representa a capacidade de o *software* estar em conformidade com normas e guias de estilos relacionados à usabilidade.

Fonte: Adaptado de ABNT (2003).

Eficiência

Este indicador representa a capacidade de o *software* se manter eficiente, com desempenho adequado para o uso, de acordo com a quantidade de recursos e funcionalidades implementados no sistema. No Quadro 4, são descritas algumas características que devem ser atendidas para este indicador.

Quadro 4. Atributos de eficiência

Atributos de eficiência	Descrição
Comportamento em relação ao tempo	Representa a capacidade de o *software* processar e executar as tarefas dentro de um intervalo adequado de tempo. Esta mesma característica deve ser identificada em relação à velocidade de transferência de informações entre sistemas.
Utilização de recursos	Representa a capacidade de o *software* utilizar quantidades adequadas de recursos ao executar tarefas.
Conformidade relacionada à eficiência	Representa a capacidade de o *software* estar em conformidade com as normas relacionadas à eficiência.

Fonte: Adaptado de ABNT (2003).

Manutenibilidade

Este indicador se refere à capacidade de o software ser mantido, ou seja, passar por correções, melhorias e adaptações relacionadas a mudanças no ambiente em que o sistema está implementado. Algumas características devem ser atendidas para este indicador, e são descritas no Quadro 5.

Quadro 5. Atributos de manutenibilidade

Atributos de manutenibilidade	Descrição
Analisabilidade	Representa a capacidade de o *software* possibilitar a identificação de falhas em sua estrutura ou a identificação de componentes que precisam ser modificados.
Modificabilidade	Representa a capacidade de o *software* possibilitar a modificação de componentes em sua estrutura, como a adição/implementação de novas funcionalidades ou adaptação de uma funcionalidade já existente.
Estabilidade	Representa a capacidade de o *software* se manter estável, evitando comportamentos inesperados após passar por modificações em sua estrutura.
Testabilidade	Representa a capacidade de o *software* possibilitar a realização de testes em sua estrutura, de modo a validar alterações que ocorreram em seus componentes.
Conformidade relacionada à manutenibilidade	Representa a capacidade de o *software* estar em conformidade com as normas e convenções vigentes relacionadas à manutenibilidade.

Fonte: Adaptado de ABNT (2003).

Portabilidade

Este indicador representa a capacidade de o *software* ser portado para outras plataformas e estruturas, sejam elas ambientes de *software* ou *hardware*, mantendo-se perfeitamente funcional. Veja no Quadro 6 algumas características que devem atender a este indicador.

Quadro 6. Atributos de portabilidade

Atributos de portabilidade	Descrição
Adaptabilidade	Representa a capacidade de o *software* se manter adaptável a mudanças de ambientes, sem a necessidade de modificações radicais na estrutura do *software*, além das já previstas.
Capacidade para ser instalado	Representa a capacidade de o *software* ser instalado em um ambiente previamente definido.
Coexistência	Representa a capacidade de o *software* coexistir em um mesmo ambiente com outros *software*, compartilhando os mesmos recursos.
Capacidade para substituir	Representa a capacidade de o *software* ser utilizado em lugar de outro sistema, atendendo às mesmas necessidades, com desempenho igual ou superior.
Conformidade relacionada à portabilidade	Representa a capacidade de o *software* estar em conformidade com as normas e convenções vigentes relacionadas à portabilidade.

Fonte: Adaptado de ABNT (2003).

Contudo, pode-se concluir que, para verificar a qualidade de um *software*, muitas métricas são utilizadas e muitas outras variáveis precisam ser consideradas. No entanto, é certo que não há uma universalidade nas métricas, podendo haver outras diferentes para diferentes cenários.

Referências

ABNT. *NBR ISO/IEC 9126-1*: engenha de software – qualidade de produto. Rio de Janeiro: ABNT, 2003.

BOEHM, B. W.; BROWN, J. R.; LIPOW, M. Quantitative evaluation of software quality. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, 2., 1976. *Proceedings* [...]. 1976. Disponível em: https://dl.acm.org/doi/10.5555/800253.807736. Acesso em: 11 dez. 2020.

BUENO, C. F. S; CAMPELO, G. B. *Qualidade de software*. Recife: UFPE, [201-?]. Disponível em: https://www.cin.ufpe.br/~mrsj/Qualidade/Qualidade%20de%20Software.pdf. Acesso em: 11 dez. 2020.

FIGUEIREDO, E. *Medição de software*. [S. l.: s. n.], 2016. Disponível em: https://homepages.dcc.ufmg.br/~figueiredo/disciplinas/2016a/dcc603/2-intro-medicao_v01.pdf. Acesso em: 11 dez. 2020.

RAMOS, R. A. *Métricas de software*: engenharia de software I. Petrolina: UNIVASF, 2005. Disponível em: http://www.univasf.edu.br/~ricardo.aramos/disciplinas/ES_I_2010_2/Aula%2005%20M%E9tricas%20de%20Software%20%5BModo%20de%20Compatibilidade%5D.pdf. Acesso em: 11 dez. 2020.

SANT'ANNA, C. N. *Manutenibilidade e reusabilidade de software orientado a aspectos*: um framework de avaliação. 2004. Dissertação (Mestrado) – PUCRIO, Rio de Janeiro, 2004. Disponível em: https://www.maxwell.vrac.puc-rio.br/colecao.php?strSecao=resultado&nrSeq=5447@1. Acesso em: 11 dez. 2020.

SILVA, B. L. R. *Qualidade de software no desenvolvimento utilizando metodologias ágeis*. 2017. 74 f. Monografia (Especialização governança em tecnologia da informação) - Instituto CEUB de Pesquisa e Desenvolvimento, Centro Universitário de Brasília, Brasília, 2017. Disponível em: http://www.aneel.gov.br/documents/656835/14876412/Especializa%C3%A7%C3%A3o+BRUNO+LOPES+2017.pdf/66610424-1b54-e1d0-d357-907df0876eb7. Acesso em: 11 dez. 2020.

Fique atento

Os *links* para *sites* da *web* fornecidos neste capítulo foram todos testados, e seu funcionamento foi comprovado no momento da publicação do material. No entanto, a rede é extremamente dinâmica; suas páginas estão constantemente mudando de local e conteúdo. Assim, os editores declaram não ter qualquer responsabilidade sobre qualidade, precisão ou integralidade das informações referidas em tais *links*.

Conteúdo:

SAGAH
SOLUÇÕES EDUCACIONAIS INTEGRADAS