UNIDADE 05

Métricas de Qualidade, Usabilidade e Manutenibilidade

META

Apresentar uma visão panorâmica das métricas de qualidade de software, com foco em usabilidade, manutenibilidade e segurança.

OBJETIVOS

Geral

Proporcionar ao aluno a compreensão dos principais modelos e métricas de qualidade de software

Específicos

- Compreender modelos de qualidade de software e seus atributos;
- Apresentar métricas aplicáveis à usabilidade;
- Apresentar métricas aplicáveis à manutenibilidade; e
- Apresentar métricas aplicáveis à segurança.

RESUMO

A qualidade de software é um conceito amplo e depende de diferentes perspectivas, especialmente da experiência do usuário. Para avaliá-la, são utilizados modelos que permitem decompor esse conceito em atributos específicos e mensuráveis. Nesta unidade, são discutidos os principais modelos de qualidade, desde os pioneiros, como o de McCall, até o modelo da ISO/IEC 25010, que organiza a qualidade em características como usabilidade, manutenibilidade e segurança.

São apresentadas métricas voltadas para cada um desses atributos. A usabilidade foi abordada como um atributo indireto, avaliado por meio de medidas como eficácia, eficiência, aprendizado e satisfação. A manutenibilidade foi discutida tanto em termos externos, associados ao processo de manutenção, quanto internos, relacionados a propriedades estruturais do software, destacando métricas como tempo médio de reparo

e número de falhas introduzidas. Já a segurança é tratada como um aspecto crítico da qualidade, com enfoque em métricas de vulnerabilidades, incidentes e no uso do *Common Vulnerability Scoring System* como ferramenta de priorização.

Em conjunto, esses conceitos e métricas oferecem uma base para compreender como medir e gerenciar a qualidade de software ao longo de seu ciclo de vida.

1. INTRODUÇÃO

O principal objetivo da engenharia de software é melhorar a qualidade dos produtos de software. Claramente, “qualidade” é uma característica que depende da perspectiva do usuário e ambiente da execução, portanto é um atributo externo. Sendo dependente do usuário, todo fator do produto do software perceptível por ele influenciará na avaliação de sua “qualidade”. Assim, características como facilidade de uso, tempo de resposta ou segurança compõem uma faceta da qualidade de um software. Nesse sentido, numa perspectiva de mensuração da qualidade, devemos defini-la em termos dos atributos relevantes para os usuários.

Nesta Unidade, discutimos modelos de qualidade de software e atributos usualmente considerados para caracterizá-la. Em seguida, discutimos medições para os atributos externos usabilidade, manutenibilidade e segurança, associados à qualidade de software.

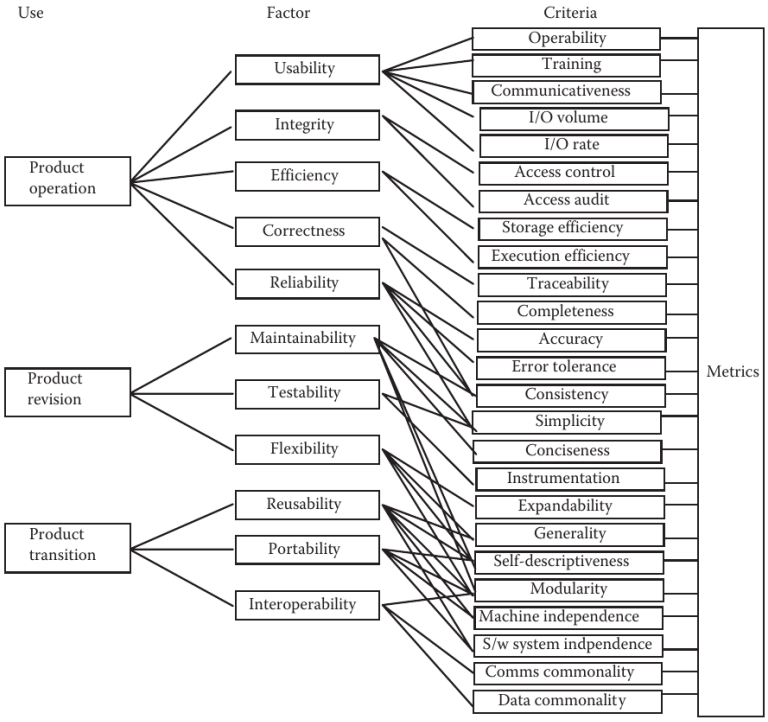
2. MODELOS DE QUALIDADE DE SOFTWARE

A qualidade de software é tipicamente representada por meio de modelos que incorporam como os diversos atributos contribuem para a qualidade geral, permitindo sua definição, decomposição e medição.

Os primeiros modelos de qualidade usam uma abordagem de decomposição. Nessa abordagem, o foco é o produto final e identificação dos atributos de qualidade numa perspectiva do usuário. Esses atributos são denominados de ***fatores de qualidade***. Tais fatores são, geralmente, atributos externos de alto nível, como confiabilidade, usabilidade e

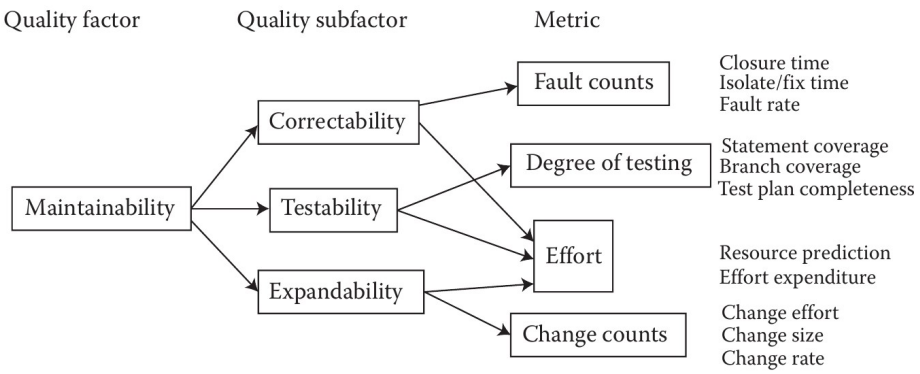
manutenibilidade, porém podem incluir atributos internos, como testabilidade. Os fatores de qualidade estão num nível muito alto de abstração para serem mensurados diretamente, assim são decompostos em outros atributos de mais baixo nível, denominados de ***critérios de qualidade*** ou ***subfatores de qualidade***. Os critérios de qualidade são decompostos em atributos que podem ser diretamente medidos, denominados de ***métricas de qualidade***. A Figura 1 apresenta o modelo de qualidade de McCall et al. (1977) nessa perspectiva e a Figura 2 ilustra a decomposição completa do fator de qualidade manutenibilidade nos critérios de qualidade e, posteriormente, nas métricas de qualidade.

Figura 1- Modelo de qualidade de software do McCall et al.



Fonte: Fenton (2014)

Figura 2- Decomposição do fator manutenibilidade em critérios e métricas.



Fonte: Fenton (2014)

SQA Research

A partir de uma evolução do modelo McCall foi definida um modelo padrão de qualidade de software, que hoje está estabelecido na ISO/IEC 25010. Nesse modelo, a qualidade é decomposta em nove características:

1. Adequação funcional;
2. Eficiência de desempenho;
3. Compatibilidade;
4. Usabilidade;
5. Confiabilidade;
6. Segurança;
7. Manutenibilidade;
8. Flexibilidade; e
9. Seguro.

SAIBA-MAIS

Na página oficial da ISO/IEC com a descrição do padrão de qualidade.

De acordo com o padrão, qualquer componente de qualidade de software pode ser expresso em termos de um ou mais dessas características. Cada uma dessas características é decomposta em outras subcaracterísticas e, posteriormente, em métricas.

Muitos engenheiros de software baseiam suas avaliações de qualidade em medidas definidas para um propósito específico, independentemente de um modelo formal de qualidade. Essas definições geralmente refletem o uso a que o software se destina ou as condições práticas de teste de um sistema. Por exemplo, determinados sistemas apresentam requisitos rigorosos de portabilidade (flexibilidade) e de integridade. O usuário ou profissional da área pode fornecer definições simples para esses termos, como:

$$Portabilidade = 1 - \frac{ET}{ER}$$

ET é uma medida da quantidade de recursos necessários para mover o sistema para outro ambiente e ER é a quantidade de recursos necessários para criar o sistema no ambiente atual.

A medição da qualidade de software por meio de abordagens baseadas em decomposição exige um planejamento cuidadoso e uma coleta de dados bem estruturada. Implementar esse processo corretamente, mesmo que para um número pequeno de atributos de qualidade, demanda recursos adicionais que, muitas vezes, os gestores não estão dispostos a disponibilizar. Em muitos casos, no entanto, basta uma avaliação aproximada da qualidade geral do software, feita a partir de dados já existentes e que não exija grandes investimentos. Por esse motivo, é comum que muitos engenheiros de software adotem uma visão mais restrita de qualidade, limitando-a apenas à ausência de defeitos no sistema. As seguintes medições podem ser usadas numa abordagem orientada a defeitos:

- Densidade de defeitos: medida da quantidade de defeitos conhecidos por tamanho do produto

$$\frac{n^o\ defeitos\ conhecidos}{tamanho\ do\ produto}$$

- Deterioração do sistema: total do tempo para corrigir um defeito em relação ao tempo total de desenvolvimento

$$\frac{custo\ para\ corrigir\ um\ defeito}{custo\ total}$$

- Taxa de remoção de defeitos: razão entre os defeitos removidos antes da entrega e por defeitos removidos totais

$$\frac{defeitos\ antes\ entrega}{defeitos\ totais}$$

- Tempo médio para detectar um defeito: média dos tempos entre a introdução de um defeito e sua detecção

3. MÉTRICAS DE USABILIDADE DE SOFTWARE

De acordo com a norma ISO/IEC 25010, usabilidade é definida como “o grau em que um produto ou sistema pode ser usado por usuários

especificados para atingir objetivos especificados com eficácia, eficiência e satisfação em um contexto de uso especificado". Essa é uma característica externa do produto, pois depende dos usuários e do contexto em que é utilizado.

A usabilidade é frequentemente entendida como "facilidade de uso" ou amigabilidade do sistema, e envolve diversos aspectos ou subatributos. Para avaliá-la, é importante verificar: quão fácil é aprender a usar o sistema, o quão eficiente o usuário consegue ser ao utilizá-lo, a facilidade de lembrar como operá-lo após algum tempo sem uso e a frequência com que ocorrem erros durante a interação. Além disso, a usabilidade também está ligada à satisfação do usuário, que é uma característica subjetiva e depende das habilidades, conhecimentos e preferências pessoais de cada indivíduo. Por isso, podemos dizer que a usabilidade é uma medida indireta, já que não pode ser avaliada apenas por um único atributo isolado.

Modelos de usabilidade baseados em atributos interno do software, como quantidade de cliques para completar um caso de uso, são preditores da usabilidade. Esses modelos não são medidas de usabilidade, pois a usabilidade só pode ser medida com o usuário incluso no contexto.

As medições mais comuns usadas para medir a usabilidade de software são:

- Efetividade: medida que indica o grau de tarefas completas realizadas pelo usuário. Essa métrica computa o percentual das tarefas que o usuário consegue realizar sem erros;
- Eficiência: envolve o tempo requerido para realizar uma tarefa. Podem ser medidos o tempo necessário para concluir uma tarefa, o número de passos ou cliques realizados e a produtividade alcançada em termos de quantidade de tarefas finalizadas por unidade de tempo;
- Satisfação: medição do grau de satisfação do usuário. Essa é uma medida subjetiva e pode ser mensurada usando escalas de satisfação ou medindo as reações biológicas (pulsação, expressões faciais) que o usuário faz ao usar o sistema;
- Aprendizado: medição do grau de facilidade de se aprender o

sistema. Pode-se medir o tempo médio para aprender a realizar uma tarefa, o número de tentativa até conseguir executar uma tarefa corretamente ou a quantidade de funções aprendidas em um determinado tempo.

4. MÉTRICAS DE MANUTENIBILIDADE DE SOFTWARE

A maioria dos softwares é utilizada repetidamente ao longo do tempo, e muitos profissionais são responsáveis por sua manutenção após a entrega. Tanto antes quanto durante esse período de manutenção, a medição de software pode trazer informações muito valiosas. Um bom software deve ser fácil de compreender, de aprimorar e de corrigir. Quando isso acontece, dizemos que o software é manutenível. Durante o desenvolvimento, algumas métricas permitem prever se esse objetivo está sendo alcançado. Após a entrega, as medições continuam importantes: elas ajudam a avaliar o impacto que uma mudança pode causar no sistema. Dessa forma, a medição apoia tanto a prevenção de problemas quanto a tomada de decisões durante a manutenção.

Manutenibilidade não é restrita a código. Ela é um atributo de diversos produtos de softwares: documentos de especificação, de projeto e de testes. Portanto, é importante medir a manutenibilidade dos diversos produtos que se deseja manter. Há duas abordagens de medição da manutenibilidade: externa e interna.

A abordagem externa mede o processo de manutenibilidade: se o processo de manutenção é efetivo, então assume-se que o produto é manutenível. Por outro lado, a abordagem interna identifica atributos interno e estabelece que são preditores do processo de manutenção. Assim como a usabilidade, manutenibilidade é um atributo externo, pois depende das pessoas que realizam a manutenção, documento e ferramentas de suporte, e o propósito de uso do software. Portanto, a abordagem interna não mede a manutenibilidade. Ela é apenas um preditor.

A manutenção está relacionada com a tarefa de realizar modificações no produto. Portanto, a velocidade com que se realiza as mudanças é uma característica importante da manutenibilidade. Assim, uma medição relevante é *MTTR (Mean Time To Repair)* que mede o tempo médio para ser realizar uma modificação. Como a manutenibilidade também está relacionada com a quantidade de mudanças, as seguintes medições podem ser realizadas ao medir a manutenibilidade de um software:

- Quantidade de problemas não resolvidos;
- Número de módulos modificados ao implementar uma mudança;
- Razão entre o tempo gasto para realizar as modificações no sistema e a quantidade de modificações realizadas;
- Percentual de modificações que introduzem novas falhas.

Na abordagem interna, inúmeras métricas já foram propostas como indicadores de manutenibilidade. Em particular, as medições de complexidade têm sido associadas com o esforço de manutenção. Isso é devido a intuição de que produtos mal estruturados e mal documentados resultarão em produtos com baixa manutenibilidade. Assim, diversas medições estruturais têm sido correlacionadas com manutenibilidade.

Considera-se que a legibilidade é uma característica importante para a manutenibilidade de produtos textuais, como código-fonte. Uma medição usada para legibilidade é o *índice F*:

$$F = 0{,}4 \times \frac{n^o palavras}{n^o sentenças} + PW3$$

Em que $PW3$ é o percentual de palavras com três ou mais sílabas dos documentos. Especificamente para código-fonte, usa-se a seguinte medição de legibilidade:

$$R = 0{,}295\,a - 0{,}499\,b + 0{,}13\,c$$

Em que a é a quantidade média de caracteres das variáveis do programa, b é a quantidade de linhas que contém comandos e c é a complexidade ciclomática do programa.

CURIOSIDADE

Sabia que algumas pesquisas estudaram o relacionamento entre padrões de projetos e probabilidade de mudanças no código e encontraram que as classes envolvidas nos padrões são mais propensas as mudanças que classes não relacionadas aos padrões?

SAIBA-MAIS

5. MÉTRICAS DE SEGURANÇA DE SOFTWARE

Veja mais detalhes de como a CVSS pode ser aplicada neste link.

A segurança de software é um dos atributos críticos da qualidade, pois se relaciona à capacidade do sistema de proteger dados e funcionalidades contra acessos não autorizados, falhas e ataques. Avaliar a segurança de um software exige a definição de métricas que forneçam informações objetivas sobre vulnerabilidades, riscos e eficácia das medidas de proteção implementadas. A medição da segurança é essencial tanto durante o desenvolvimento quanto ao longo do ciclo de vida do software, pois permite identificar pontos fracos antes da entrega e monitorar a confiabilidade do sistema durante sua operação.

O risco de segurança de uma organização ou sistema é calculado pelos produtos dos fatores de impacto, probabilidade, ameaça e vulnerabilidade. Esses fatores são interdependentes e ao atribuir o risco de segurança é necessário balanceá-los adequadamente. O método mais usado para balancear os múltiplos fatores é o *CVSS (Common Vulnerability Scoring System)*. A métrica base CVSS estabelece um valor de 0 a 1, em que 0 indica nenhuma vulnerabilidade e 1 máxima vulnerabilidade. Ela apresenta seis medições base relacionadas com o acesso requerido para explorar a vulnerabilidade e o seu impacto:

- Acesso ao vetor (AV): indica o nível de acesso a rede que o atacante precisa ter para explorar a vulnerabilidade;
- Complexidade de acesso (AC): indica a complexidade de ataque para conseguir ter sucesso;
- Autenticação (AU): indica quantas vezes o atacante precisa autenticar para ter sucesso;
- Confidencialidade (C): indica o nível de impacto na confidencialidade dos dados caso o atacante tenha sucesso;
- Integridade (I): indica o nível de impacto na integridade do sistema; e
- Disponibilidade (A): indica o nível de impacto na disponibilidade do sistema.

Cada uma dessas medições é realizada numa escala ordinal de nível baixo, médio e alto com atribuições padrão de valores numéricos para cada um. Além dessas métricas bases, CVSS incluem métricas temporais e de ambiente. As métricas temporais como os atributos de vulnerabilidade evoluem e são computados por especialistas em segurança e disponíveis em sites. As métricas de ambientes são dependentes das características individuais do sistema e medidas pelos desenvolvedores ou usuários mais familiarizados com os possíveis impactos das possíveis vulnerabilidades do sistema.

6. CONSIDERAÇÕES FINAIS

Nesta unidade, discutimos a importância da mensuração na avaliação da qualidade de software, destacando que esse atributo é multidimensional e depende tanto da perspectiva do usuário quanto do contexto de uso. Foram apresentados os principais modelos de qualidade, desde a abordagem de decomposição proposta por McCall até o modelo mais recente da ISO/IEC 25010, que organiza a qualidade em oito características fundamentais.

Na sequência, exploramos métricas específicas para três atributos externos relevantes: usabilidade, manutenibilidade e segurança. Em usabilidade, enfatizamos aspectos como eficácia, eficiência, satisfação, e aprendizado, ressaltando seu caráter indireto e dependente do usuário. Em manutenibilidade, abordamos tanto a perspectiva externa, ligada ao processo de manutenção, quanto a interna, com métricas de estrutura e legibilidade de código, destacando indicadores como MTTR e a taxa de falhas introduzidas. Por fim, tratamos da segurança como um atributo essencial, discutindo métricas de vulnerabilidade, incidentes e a aplicação do CVSS (Common Vulnerability Scoring System) como padrão para avaliar a gravidade de falhas.

REFERÊNCIAS

FENTON, Norman E.; BIEMAN, James. Software Metrics: a Rigorous

and Practical Approach. 3. ed. Boca Raton: CRC Press, 2014.

McCall J.A., Richards P.K., and Walters, G.F., Factors in software quality, RADC TR-77-369, 1977. Vols I, II, III, US Rome Air Development Center Reports NTIS AD/A-049 014, 015, 055, 1977.