

Software Engineering Body of Knowledge (SWEBOk)

Apresentação

Mesmo em assuntos não ligados à tecnologia, cada vez mais se faz necessária a padronização e o uso de métodos e processos para melhorar a qualidade dos projetos. Nesse sentido, a IEEE desenvolveu e mantém o guia SWEBOk com a intenção de divulgar as melhores práticas relacionadas à engenharia de software.

O Guia SWEBOk, em sua terceira versão, é composto por quinze áreas do conhecimento que abrangem os principais aspectos relacionados à projetos de engenharia de software. Cada uma dessas áreas foram desenvolvidas por meio de boas práticas coletadas ao longo de décadas e em vários países.

Nesta Unidade de Aprendizagem, você vai conhecer o guia SWEBOk, bem como seus objetivos, áreas de conhecimento e informações relacionadas às certificações profissionais disponíveis.

Bons estudos.

Ao final desta Unidade de Aprendizagem, você deve apresentar os seguintes aprendizados:

- Descrever o guia SWEBOk e seus objetivos.
- Categorizar as áreas de conhecimento do SWEBOk.
- Explicar as modalidades de certificação do SWEBOk.

Desafio

A análise de requisitos expressa as necessidades e restrições colocadas sobre o produto, as quais contribuem para a solução de algum problema do mundo real. Algumas das atividades importantes para a análise de requisitos são a especificação e a validação dos requisitos de software.

Baseando-se nesse contexto, considere o seguinte cenário:

Você é o gestor de TI de uma pequena empresa e está participando de uma reunião com um cliente que deseja comprar um *software* para gerenciamento de sua escola.

Veja mais sobre esse caso:

O SOFTWARE PARA GERENCIAMENTO DE ESCOLA

O cliente comentou, em reunião, que são ofertados diversos cursos, cada um com um professor coordenador, que contém disciplinas diversas. As disciplinas têm uma carga horária pré-determinada, e precisam apresentar dados relacionados a sua ementa, bibliografia, e quais disciplinas são pré-requisitos. Para fazer as matrículas junto à secretaria, os alunos precisam ser cadastrados com nome, endereço, um código de matrícula para manter o controle e o curso que estão fazendo. Os alunos também precisam de acesso ao sistema, para consulta de notas. Além disso, é necessário manter o registro dos professores que vão ministrar as disciplinas aos alunos, com dados adicionais como telefone, titulação e cursos vinculados. Agora que a reunião terminou, você precisa elencar os requisitos funcionais identificados na reunião.



Agora que a reunião terminou, você precisa elencar os requisitos identificados na reunião.

Tabela de requisitos

Requisito	Descrição

1. Preencha a tabela com os requisitos levantados.
2. Sinalize qual é a área de conhecimento necessária para cumprir essa etapa do projeto.

Infográfico

Cada vez mais se faz necessária a produção de qualidade e que ela satisfaça os clientes e os padrões de qualidade das organizações. Para isso, a engenharia de software tem evoluído com modelos e processos de desenvolvimento descritos em corpos de conhecimento.

Assim como o SWEBOK é um guia para a engenharia de software, o PMBOK é um importante guia para o gerenciamento de projetos.

Veja um comparativo entre SWEBOK e PMBOK neste Infográfico.

SWEBOK X PMBOK

Veja a diferença entre os guias SWEBOK e PMBOK.



SWEBOK

- ▶ Guia com assuntos essenciais na área de engenharia de software.
- ▶ É conduzido pelo IEEE.
- ▶ Tem como objetivo principal estabelecer um conjunto apropriado de critérios e normas para a prática profissional da engenharia de software.
- ▶ É estruturado em áreas de conhecimento.
- ▶ Descreve processos de desenvolvimento e manutenção de software.
- ▶ É aplicável na área de engenharia de software.
- ▶ É pouco utilizado no Brasil.



PMBOK

- ▶ Conjunto de práticas de gerência de projetos.
- ▶ Conduzido pelo PMI.
- ▶ Tem como objetivo principal fornecer uma visão geral sobre o conjunto de conhecimentos em gerenciamento de projetos.
- ▶ É estruturado em áreas de conhecimento e grupos de processos.
- ▶ Descreve processos, ferramentas e técnicas de gerenciamento de projetos.
- ▶ É aplicável em várias áreas da organização.
- ▶ É muito utilizado no Brasil.

Certificações relacionadas:

- ▶ Certificação de Desenvolvedor de Software Associado (ASD);
- ▶ Certificação para Desenvolvedor de Software profissional (PSD).
- ▶ Mestre Profissional em Engenharia de Software (PSEM)

Certificações relacionadas:

- ▶ Técnico Certificado em Gestão de Projetos (CAPM)
- ▶ Profissional de Gestão de Projetos (PMP)
- ▶ Profissional de Gestão de Programas (PgMP)
- ▶ Profissional de Gestão de Portfólio (PfMP)
- ▶ Profissional em Análise de Negócios do PMI (PMI-PBA)
- ▶ Profissional Ágil Certificado do PMI (PMI-ACP)
- ▶ Profissional em Gerenciamento de Riscos do PMI (PMI-RMP)
- ▶ Profissional de Gestão de Cronograma PMI (PMI-SP)



Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.

Conteúdo do Livro

Uma das principais referências na engenharia de software é o SWEBOk. Este guia descreve conceitos e práticas relacionados à engenharia de software, e a divide em áreas de conhecimento, que auxiliam na entrega de produtos de software de qualidade.

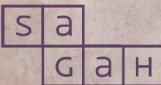
As áreas de conhecimento do SWEBOk percorrem todo o processo de desenvolvimento de software, incluindo processos para a gerência dos projetos de software, passando pela coleta inicial de requisitos, design, testes, desenvolvimento, manutenção e demais assuntos relacionados. Desta forma, o SWEBOk pode ser utilizado como guia para os mais diversos projetos e processos relacionados à engenharia de software.

No capítulo Software Engineering Body of Knowledge - SWEBOk, da obra *Engenharia de software*, você aprenderá sobre as diferentes áreas de conhecimento e sobre as certificações relacionadas ao SWEBOk.

Boa leitura.

ENGENHARIA DE SOFTWARE

Adriana de Souza Vettorazzo



SOLUÇÕES
EDUCACIONAIS
INTEGRADAS

Software Engineering Body of Knowledge (SWEBOk)

Objetivos de aprendizagem

Ao final deste texto, você deve apresentar os seguintes aprendizados:

- Descrever o guia SWEBOk e seus objetivos.
- Categorizar as áreas de conhecimento do SWEBOk.
- Explicar as modalidades de certificação sobre o SWEBOk.

Introdução

Com o aumento da demanda pelo desenvolvimento de *software*, em decorrência do crescente número de aplicativos disponíveis em dispositivos móveis, surgiu também a necessidade de padronização e métodos para melhorar a qualidade desses *software*. Nesse sentido, foi elaborado o *Software Engineering Body of Knowledge* (SWEBOk), um guia de uso e aplicação das melhores práticas de engenharia de *software*.

O SWEBOk foi desenvolvido a partir de boas práticas coletadas ao longo de décadas em vários países. Seu principal objetivo é divulgar um conjunto de práticas, ferramentas e técnicas para o desenvolvimento de atividades de engenharia de *software*.

Assim, neste capítulo, você vai estudar o SWEBOk, compreendendo quais são seus objetivos e suas áreas de conhecimento. Por fim, você vai verificar quais são as modalidades de certificação profissional relacionadas a esse guia.

1 Projeto SWEBOk

Segundo Pressman e Maxim (2016), a **engenharia de software** é uma disciplina que reúne um conjunto de métodos, processos e ferramentas para o desenvolvimento de *software*, com processos que podem ser utilizados em todo o ciclo de vida de um projeto de *software*. Nesse contexto, o Instituto de

Engenheiros Eletricistas e Eletrônicos (IEEE, do inglês Institute of Electrical and Electronics Engineers) é responsável pelo projeto do **SWEBOK**, que atualmente está em sua terceira versão e é disponibilizado de forma gratuita no *website* da IEEE Computer Society (BOURQUE; FAIRLEY, 2014). O SWEBOK tem o objetivo de auxiliar as organizações que trabalham com desenvolvimento de sistemas a terem uma visão mais consistente sobre o processo de desenvolvimento de *software*.

O principal objetivo do guia SWEBOK é descrever o conhecimento relacionado à área da engenharia de *software*. O guia também é conhecido como Relatório Técnico 19759 da Organização Internacional de Normalização (ISO, do inglês International Organization for Standardization) e da Comissão Eletrotécnica Internacional (IEC, do inglês International Electrotechnical Commission). Além disso, segundo a IEEE (c2020, tradução nossa), o guia SWEBOK:

- apresenta e descreve as características relacionadas ao conteúdo da engenharia de *software*;
- promove uma visão consistente de como a engenharia de *software* é vista e utilizada no âmbito mundial;
- esclarece e delimita as fronteiras entre a engenharia de *software* e as outras disciplinas relacionadas (ciência da computação, gerência de projetos, engenharia da computação, matemática, entre outras);
- disponibiliza material de apoio para o autoaprendizado e a certificação individual de engenheiros de *software*.

A versão 3 do SWEBOK, lançada em 2014, é composta por 15 áreas de conhecimento, também conhecidas por *knowledge areas*. São elas:

1. Requisitos de *software*
2. *Design* de *software*
3. Construção de *software*
4. Teste de *software*
5. Manutenção de *software*
6. Gerência de configuração de *software*
7. Gerência de engenharia de *software*
8. Processo de engenharia de *software*
9. Modelos e métodos de engenharia de *software*
10. Qualidade de *software*
11. Prática em engenharia de *software* profissional

12. Economia em engenharia de *software*
13. Fundamentos da computação
14. Fundamentos matemáticos
15. Fundamentos da engenharia

A seguir, vamos explicar cada uma das 15 áreas.

1 Requisitos de *software*

Os **requisitos** correspondem às necessidades e restrições do produto de *software* e ajudam nas soluções e nos problemas do mundo real. Nessa área, temos a elicitação, a análise, as especificações e a validação dos requisitos funcionais e não funcionais. Nos projetos de *software*, na maioria das vezes, as principais falhas ocorrem devido às dificuldades em se entender as necessidades do usuário. Por isso, fazer um bom levantamento de requisitos é extremamente importante. O SWEBOk apresenta alguns pontos relacionados à área de requisitos de *software*, descritos a seguir.

- Fundamentos dos requisitos de *software*: requisitos são propriedades que devem espelhar o mundo real com o intuito de resolver um problema. Requisitos podem ser de produto ou de processo, funcionais ou não funcionais, e de sistema ou de *software*. Também podem apresentar propriedades emergentes. Precisam ser, preferencialmente, quantificáveis.
- Processo de requisitos: essa área inclui o planejamento de requisitos, mostrando como os processos de requisitos vão integrar-se com os processos de *software*. Nessa fase, são discutidos os modelos e atores de processo. Também é definido como serão o suporte e o gerenciamento de processos, além dos processos que visam à qualidade e à melhoria de processos.
- Elicitação de requisitos: essa área visa a efetuar a captura e a aquisição dos requisitos de *software*, identificando as fontes dos requisitos. Nessa fase, é importante definir as fontes de requisitos e as técnicas de elicitação.
- Análise de requisitos: a análise é a responsável por fazer a ligação entre a alocação de *software* em nível de sistema e o projeto de *software*, ajudando o engenheiro a aprimorar e construir modelos. Segundo Pressman e Maxim (2016, p. 122), “[...] se você não analisa, é altamente provável que construa uma solução de *software* muito elegante que resolve o problema errado”. Nessa fase, são feitas a classificação e a modelagem

conceitual de requisitos, o projeto arquitetural e a alocação de requisitos, a negociação de requisitos entre o time de desenvolvimento e o cliente e, por fim, a análise formal.

- Especificação de requisitos: compreende a atribuição de valores numéricos para os objetivos do projeto, e a atividade principal dessa área é a documentação do sistema. Nessa fase, é feita a documentação de definição do sistema, a especificação dos requisitos de sistema e a especificação dos requisitos de *software*.
- Validação de requisitos: a documentação levantada na especificação de requisitos pode ser utilizada para a validação da conformidade com os padrões da organização. Essa fase visa a enumerar os problemas encontrados e inclui a revisão de requisitos, a prototipação, a validação de modelos e os testes de aceitação.
- Considerações práticas: aqui são descritos tópicos para serem compreendidos na prática e, ao mesmo tempo, para validar os atributos e o tamanho das alterações nos requisitos e fazer a estimativa de custo para o desenvolvimento e a manutenção
- Ferramentas para requisitos de *software*: usualmente suportam diversas atividades, como documentação, rastreamento e gerenciamento da mudança. Há duas principais categorias: ferramentas para modelagem e ferramentas para o gerenciamento de requisitos.

2 Design de *software*

A área do **design, ou projeto, de software** está relacionada ao ciclo de vida do projeto, em que são feitas a análise dos requisitos e a descrição da estrutura interna do *software*. Aqui, definem-se a arquitetura, os componentes, as relações e outras características do sistema. Essa área é apresentada com a estrutura descrita a seguir.

- Fundamentos do projeto de *software*: discute os conceitos gerais relacionados à arquitetura do *software*, como o contexto, o processo e os princípios fundamentais em *design de software*.
- Questões-chave em *design de software*: nessa fase, são discutidas questões relacionadas à arquitetura, como concorrência, controle de eventos, persistência de dados e segurança.
- Estrutura e arquitetura de *software*: apresenta estruturas e estilos arquiteturais e padrões, discute as decisões de arquitetura e apresenta conjuntos de programas e *frameworks*.

- Projeto da interface com o usuário: discursa sobre questões relacionadas a princípios, problemas, modalidades, formas de apresentação e outros tópicos relacionados à experiência do usuário com o sistema.
- Análise e avaliação de qualidade do projeto de *software*: discute os atributos de qualidade, a análise, as métricas e as técnicas de avaliação de *design de software*.
- Notações do projeto de *software*: diferencia descrições estruturais (visualizações estáticas) e descrições comportamentais (visualizações dinâmicas).
- Estratégias e métodos para o projeto de *software*: apresenta as principais estratégias voltadas ao *design de software*, como orientada a função, orientada a objeto, centrada nos dados ou baseada em componentes, entre outras.
- Ferramentas para o projeto de *software*: podem ser utilizadas como apoio para a criação do artefato de *software*. Usualmente, possibilitam as atividades de traduzir o modelo de requisitos em uma representação de projeto, além de apoiar a representação da interface e dos componentes funcionais, entre outras atividades.

3 Construção de *software*

Essa área integra as demais áreas de conhecimento do guia SWEBOk, mas se destaca um relacionamento maior com o projeto e os testes de *software*. Estão inseridas aqui as atividades de implementação, verificação, testes de unidade e teste de integração e depuração. Essa seção trata dos pontos descritos a seguir.

- Fundamentos da construção: apresenta técnicas para ajudar a reduzir a complexidade e antecipar necessidades de mudança, verificação, reúso e outros padrões relacionados.
- Gerenciamento da construção: introduz os modelos de ciclo de vida, o planejamento e as métricas relacionadas à construção de *software*.
- Considerações práticas: trata de questões como o *design* da construção, as linguagens, a codificação, a qualidade na construção, a integração, entre outros tópicos relacionados.
- Tecnologias para construção: apresenta diversos conceitos relacionados a tecnologias para o desenvolvimento de *software*, como projeto e uso de interfaces de programação de aplicações, padrões de plataforma, modelos executáveis, *middleware* etc.

- Ferramentas para a construção de *software*: apoiam o desenvolvimento do *software* em si por meio dos ambientes de desenvolvimento, dos construtores de interface gráfica, das ferramentas para teste unitário e das ferramentas para análise de desempenho.

4 Teste de *software*

Segundo o SWEBOK (BOURQUE; FAIRLEY, 2014), o **teste de *software*** é uma atividade realizada para a avaliação da qualidade do produto, efetuando sua melhoria por meio da identificação de defeitos e problemas. Aqui, é feita a verificação do comportamento do *software* a partir de casos de testes, para verificar se o comportamento está dentro do esperado. A área de testes de *software* inclui as subáreas descritas a seguir.

- Fundamentos de teste de *software*: introduz terminologias e problemas-chave e fala do relacionamento de testes de *software* com outras áreas do conhecimento.
- Níveis de teste: discute a importância de definir bem o alvo e os objetivos do teste.
- Técnicas de testes: apresenta as técnicas mais utilizadas, como a técnica baseada na intuição e experiência do engenheiro de *software*, a baseada no código e a baseada em falhas, e discute como selecionar e/ou combinar as técnicas mais adequadas de acordo com o contexto.
- Métricas relacionadas aos testes: trata especificamente de duas avaliações, a avaliação do programa que está sendo testado e a avaliação dos testes que foram executados.
- Processo de testes: fala das aplicações práticas relacionadas às atividades de testes.
- Ferramentas para o teste de *software*: inclui ferramentas para a criação de *drivers* e *stubs*, para a geração de casos de teste, para a captura e repetição de erros, para testes de regressão, entre outras.

5 Manutenção de *software*

Essa área tem como foco principal dar suporte ao produto no **ciclo de vida operacional**. Esse suporte efetivo pode ser fornecido antes ou depois da entrega ao cliente: antes, desenvolvendo atividades de planejamento; depois, nas codificações para corrigir as falhas e melhorar o desempenho. De forma

geral, as manutenções de *software* podem ser corretivas, adaptativas, perfeitas ou preventivas.

Na **manutenção corretiva**, o *software* é modificado para corrigir os erros encontrados. Na **manutenção adaptativa**, o *software* é alterado para se adaptar às mudanças do ambiente externo. Já na **manutenção perfectiva**, o *software* é aprimorado conforme solicitação do cliente. Por fim, na **manutenção preventiva**, o *software* é modificado para facilitar as correções, adaptações e melhorias.

A área de manutenção é apresentada conforme as subáreas apresentadas a seguir.

- Fundamentos de manutenção de *software*: introduz a terminologia relacionada à manutenção de *software*, à natureza da manutenção, à sua necessidade, além de categorias, custos etc.
- Questões-chave em manutenção de *software*: discute questões técnicas, gerenciais e relacionadas à estimativa de custos da manutenção.
- Processo de manutenção: especifica as práticas mais comuns relacionadas a processos e atividades de manutenção.
- Técnicas para a manutenção: apresenta técnicas como reengenharia, engenharia reversa e migração.
- Ferramentas para a manutenção de *software*: ferramentas que são utilizadas quando o *software* está sendo modificado, como ferramentas que verificam a dependência entre trechos de código-fonte.

6 Gerência da configuração de *software* (GCS)

Essa é a área que tem por finalidade controlar as versões do *software* durante o desenvolvimento do projeto e apresenta as atividades descritas a seguir.

- Gerenciamento do processo de GCS: discute o contexto organizacional, as restrições e guias para o processo de GCS, o planejamento, o plano e a sobrevivência de sistemas de GCS.
- Identificação da configuração do *software*: apresenta os itens a serem controlados e as bibliotecas de *software*.
- Controle da configuração do *software*: fala sobre requisições, avaliações e aprovação de mudanças, sobre implementação de mudanças nos *software* e sobre desvios e desistências.
- Controle do *status* de configuração de *software*: apresenta a importância da informação e do relatório de *status* do *software* de configuração.

- Auditoria da configuração de *software*: discute a auditoria funcional e física.
- Gerenciamento e entrega das *releases* do *software*: discute as entregas de *software* e o versionamento.
- Ferramentas para a GCS: incluem ferramentas para a identificação, o empacotamento e a entrega do produto de *software*. Ferramentas nessa categoria incluem ferramentas de controle de versão e de controle de mudanças. Um exemplo muito utilizado é o GitHub (c2020), uma das ferramentas de versionamento de *software* mais utilizadas no mundo, que hospeda mais de 100 milhões de repositórios de código-fonte contendo as mais diferentes linguagens de programação.

7 Gerência da engenharia de *software*

Esta área se destina às atividades de gerenciamento do projeto de *software*, como planejamento, coordenação, medição, monitoração, controle e documentação, a fim de garantir *software* de qualidade. O projeto é algo com início, meio e fim, ou seja, é temporário e tem o objetivo de criar um produto ou um serviço.

Segundo Pressman e Maxim (2016), a gestão do projeto engloba o planejamento, a monitoração e o controle de todos os eventos que acontecem durante o processo de desenvolvimento do *software*. O gerenciamento da engenharia de *software* é dividido conforme descrito a seguir.

- Iniciação e definição do escopo: inclui a determinação e negociação dos requisitos, a análise de viabilidade e a discussão do processo para revisão de requisitos.
- Planejamento do projeto de *software*: trata do planejamento, da determinação dos entregáveis do projeto, das estimativas de esforço, do cronograma e custo, da alocação de recursos, da gerência de riscos, da gerência da qualidade e da gerência do plano do projeto.
- Declaração do projeto de *software*: inicia a implementação do plano definido na etapa anterior. Trata da aquisição de recursos, da implementação de métricas de processo, do monitoramento, do controle e dos relatórios do progresso.
- Revisão e avaliação: determina a satisfação dos requisitos e revisa e avalia o desempenho.
- Encerramento: determina o fechamento do projeto e encerra as atividades.

- Métricas de desempenho em engenharia de *software*: estabelece e mantém um comprometimento com as medições, planeja e executa os processos de medição e, por fim, avalia as medições.
- Ferramentas para o gerenciamento da engenharia de *software*: facilitam a visibilidade e o controle do *software* em desenvolvimento. Incluem ferramentas para planejamento e rastreabilidade do projeto, para gerenciamento de riscos, para facilitar a comunicação e para cálculo das métricas.

8 Processo de engenharia de *software*

Compreende um conjunto de atividades que facilitam a transformação de ideias abstratas em projetos de *software*. É composto pelas subáreas descritas a seguir.

- Definição do processo de *software*: definições gerenciais e arquiteturais.
- Ciclo de vida do *software*: apresenta categorias, modelos, adaptação de processos e outras considerações práticas.
- Avaliação e melhoria do processo de *software*: trata sobre os modelos e métricas relacionados à avaliação e sobre os modelos relacionados à melhoria do processo de *software*.
- Métricas de *software*: métricas relacionadas a processo e produto, à qualidade dos resultados mensurados, aos modelos de informação de *software* e às técnicas para identificar métricas de processo de *software*.
- Ferramentas para o processo de engenharia de *software*: inclui ferramentas para desenho de fluxos, diagramas, mapeamento das atividades, painel de controle, *dashboard* etc.

9 Modelos e métodos de engenharia de *software*

Nessa área, são pesquisados modelos e métodos que aumentem a produtividade, auxiliando no ciclo de vida do *software*. Essas ferramentas automatizam atividades do processo, liberando o analista para se concentrar nas atividades que exigem esforço intelectual. Essa área se divide nas subáreas descritas a seguir.

- Modelagem: introduz os princípios da modelagem, as propriedades e a expressão dos modelos, contextualiza sintaxe, semântica e pragmática, além de pré-condições, pós-condições e invariantes.

- Tipos de modelos: discute os modelos informacional, comportamental e estrutural.
- Análise de modelos: discursa sobre a completeza, a consistência, a correção, a rastreabilidade e a análise de interação dos modelos.
- Métodos de engenharia de *software*: discute os métodos heurístico, formal, por protótipo e ágeis.

10 Qualidade de *software*

Para falar de qualidade, é necessário observar vários aspectos relacionados ao produto, como desenvolvimento, manutenção e uso. Melhorar a qualidade é um dos principais objetivos da engenharia de *software*, e isso é possível a partir da utilização de métodos e tecnologias.

Empresas que trabalham a melhoria da qualidade geralmente utilizam modelos como Capability Maturity Model Integration ou normas técnicas como a ISO/IEC 9126. Essas empresas podem pleitear a certificação como forma de garantir aos seus clientes que elas desenvolvem e entregam produtos com maior qualidade. A qualidade de *software* se divide nas subáreas descritas a seguir.

- Fundamentos da qualidade de *software*: discute o impacto da cultura e da ética na qualidade de *software*, a relação de valor *versus* custo na qualidade, as características de modelos, a melhoria na qualidade do *software* e a segurança de sistemas.
- Processos de gerência da qualidade de *software*: apresenta os conceitos relacionados à garantia da qualidade, à verificação, à validação, às revisões e à auditoria da qualidade do *software*.
- Considerações práticas: fala sobre os requisitos de qualidade, introduz a caracterização de defeito, apresenta técnicas para a gerência da qualidade do *software* e discursa sobre as métricas de qualidade no *software*.
- Ferramentas para o processo de qualidade de *software*: inclui ferramentas de análise estática e dinâmica para analisar a qualidade do *software*.

As primeiras 10 áreas de conhecimento, discutidas até então, tratam especificamente de conteúdo diretamente relacionado ao produto e ao processo de *software*. As próximas cinco áreas a serem apresentadas também são bastante relevantes para a engenharia de *software*, pois tratam de conceitos relacionados e fundamentais à prática da engenharia de *software*.

11 Prática profissional em engenharia de software

Esta área apresenta os conhecimentos, as habilidades e as atitudes necessárias a um indivíduo que trabalha com engenharia de *software*. Discursa sobre questões como o profissionalismo e o uso de dinâmicas de grupo e psicologia e fala da importância das habilidades de comunicação.

12 Economia em engenharia de software

Aborda o contexto econômico da engenharia de *software* pela perspectiva de negócios, com bastante ênfase no custo para o desenvolvimento de um projeto de *software*. Nessa área, são apresentados os fundamentos de economia em engenharia de *software*. Também se discute o ciclo de vida da economia, os riscos e as incertezas, os métodos de análise econômica e, por fim, as considerações práticas em relação ao aspecto econômico para o desenvolvimento de sistemas.

13 Fundamentos de computação

Na área de conhecimento sobre os fundamentos de computação, são abordados conceitos fundamentais relacionados à ciência da computação, como as principais técnicas para resolução de problemas, a abstração, os fundamentos de programação, as ferramentas e técnicas para debugação, as noções básicas sobre linguagem de programação, os fatores humanos etc.

14 Fundamentos matemáticos

Apresenta conceitos matemáticos importantes para facilitar a abstração de engenheiros de *software* ao trabalhar com um problema computacional baseado em matemática, como a teoria dos conjuntos, as relações e funções, a lógica básica, os grafos e árvores, a teoria dos números, as estruturas algébricas, entre outros.

15 Fundamentos de engenharia

Entre as técnicas e os métodos de engenharia que podem ser aplicadas à engenharia de *software*, podemos citar: métodos empíricos e técnicas experimentais, análise estatística, métricas, modelagem, simulação e prototipagem, análise de causa raiz, entre outras.

2 Certificações SWEBOk

O mercado de tecnologia da informação vem desenvolvendo-se e tornando-se cada vez mais competitivo no que diz respeito a novas tecnologias e também aos serviços prestados. Adotar modelos e padrões de qualidade ajuda as empresas a se diferenciarem das demais, pois, com isso, demonstram maior organização e preocupação quanto aos produtos desenvolvidos. A IEEE disponibiliza três tipos de certificação relacionadas ao SWEBOk, descritas a seguir.

- Associate Software Developer — Certificação de Desenvolvedor de *Software* Associado: para esta certificação, os candidatos devem comprovar que possuem os requisitos básicos de conhecimento relacionados ao desenvolvimento de produtos de *software*. O candidato deve demonstrar conhecimento sobre os princípios e processos relacionados às áreas de requisitos, *design*, construção e teste de *software*.
 - Não há pré-requisitos para esta certificação, embora seja essencial o estudo do SWEBOk.
 - O exame é *on-line*, dura 100 minutos e é composto por 80 questões.
- Professional Software Developer — Certificação para Desenvolvedor de *Software* Profissional: o candidato a esta modalidade deve apresentar proficiência como desenvolvedor de *software* nas mesmas quatro áreas de conhecimento básicas (requisitos, *design*, construção e teste de *software*).
 - Não há pré-requisitos para esta certificação, embora seja essencial o estudo do SWEBOk. Ainda, é recomendado um mínimo de dois anos de educação superior em Ciência da Computação ou equivalente, além de dois anos de experiência na indústria de *software*.
 - O exame é *on-line*, dura três horas e é composto por 160 questões.
- Professional Software Engineering Master — Mestre Profissional em Engenharia de *Software*: esta é a certificação mais completa e mais complexa. O candidato deverá comprovar maestria em 11 áreas de conhecimento do SWEBOk: requisitos, *design*, construção, teste, manutenção, gerência de configuração, gerência da engenharia de *software*, modelos e métodos, qualidade e economia em engenharia de *software*.
 - Não há pré-requisitos para esta certificação, embora seja essencial o estudo do SWEBOk. Ainda, é recomendado um mínimo de quatro anos de educação superior em Ciência da Computação ou equivalente, além de quatro anos de experiência na indústria de *software*.
 - O exame é *on-line*, dura três horas e é composto por 160 questões.



Saiba mais

As certificações do desenvolvedor de software associado da IEEE Computer Society são projetadas para avaliar e validar o conhecimento de engenharia de software e as habilidades de desenvolvimento. A avaliação reúne várias áreas de conhecimento inter-relacionadas, para avaliar a capacidade de um candidato em compreender os conceitos, integrar várias áreas de conhecimento e aplicá-las na prática. Para saber sobre valores e inscrição, acesse o site da IEEE (IEEE, c2020).



Referências

BOURQUE, P.; FAIRLEY, R. E. (ed.). *SWEBOk v3. 0: Guide to the software engineering body of knowledge*. Washington: IEEE Computer Society, 2014.

GITHUB, c2020. Disponível em: <https://github.com/about>. Acesso em: 22 out. 2020.

IEEE COMPUTER SOCIETY. *Software Engineering Body of Knowledge (SWEBOk)*, c2020. Disponível em: <https://www.computer.org/education/bodies-of-knowledge/software-engineering>. Acesso em: 22 out. 2020.

PRESSMAN, R. S.; MAXIM, B. R. *Engenharia de software uma abordagem profissional*. 8. ed. Porto Alegre: Bookman, 2016.

Leitura recomendada

SOMMERVILLE, I.; SOMMERVILLE, I. *Engenharia de software*. 9. ed. São Paulo: Pearson, 2011.



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*.

Encerra aqui o trecho do livro disponibilizado para esta Unidade de Aprendizagem. Na Biblioteca Virtual da Instituição, você encontra a obra na íntegra.

Conteúdo:



SOLUÇÕES
EDUCACIONAIS
INTEGRADAS

Dica do Professor

Para manter-se dentro das exigências de mercado, é necessário buscar conhecimento e melhorar habilidades enquanto profissional de TI. As certificações são grandes aliadas para destacar o currículo no mercado de trabalho, a exemplo das certificações do SWEBOK.

Nesta Dica do Professor, você irá acompanhar algumas dicas de como obter sucesso nas provas de certificação.



Aponte a câmera para o código e accese o link do conteúdo ou clique no código para accesar.

Exercícios

- 1) O SWEBOK apresenta a engenharia de *software* como um conjunto de áreas de conhecimento necessários para se trabalhar com desenvolvimento de *software*, desde a codificação em si até a gestão de projetos. Dentre essas áreas, há uma delas que aponta o gerenciamento e a mensuração da engenharia de *software*, outra que é composta pela verificação dinâmica de uma seleção de domínios de execuções, normalmente infinito, contra o comportamento esperado e, além destas, uma área que aborda considerações relativas à qualidade de *software*, que vão além dos processos de ciclo de vida dele.

Analise as opções abaixo e assinale as que correspondem às três áreas citadas.

- A) Gerência de engenharia de *software*, teste de *software*, qualidade de *software*.
- B) Processo de *software*, manutenção de *software*, processo de negócio de *software*.
- C) Projeto de *software*, processo de negócio de *software*, gerência de projeto de *software*.
- D) Gerência de projeto de *software*, gerência de engenharia de *software*, teste de *software*.
- E) Ferramentas e métodos da engenharia de *software*, teste de *software*, qualidade de *software*.
- 2) A engenharia de *software* é uma área do conhecimento da computação voltada para especificação, desenvolvimento e manutenção de sistemas de *software*, aplicando tecnologias e práticas de gerência de projetos e outras disciplinas, objetivando organização, produtividade e qualidade.

Visando esses objetivos, a IEEE desenvolveu o SWEBOK, que é um guia de referência organizado e que contém um conjunto de conhecimentos que foram divididos em áreas e subáreas.

Baseado nesse contexto, analise as alternativas abaixo e assinale a que representa a quantidade de áreas do guia SWEBOK, 2014.

- A) 5 áreas
- B) 6 áreas
- C) 11 áreas

- D) 9 áreas
- E) 15 áreas
- 3) Qualidade de *software* aborda considerações relativas à qualidade que vão além dos processos de ciclo de vida dele. Uma vez que a qualidade de *software* é um assunto presente em todas as partes na engenharia de *software*, também é considerada em muitas outras áreas de conhecimento. A área do conhecimento *Qualidade de Software*, do SWEBOk, está dividida em tópicos. São eles:
- Fundamentos de qualidade de *software*.
 - Métricas de desempenho.
 - Gerência do processo de qualidade de *software*.
 - Considerações práticas.
- Analise as opções a seguir e assinale a que corresponde aos tópicos da área de qualidade de *software*.
- A) Estão corretas as alternativas b e c.
- B) Estão corretas as alternativas a, b e c.
- C) Estão corretas as alternativas a, b e d.
- D) Estão corretas as alternativas a, c e d.
- E) Estão corretas as alternativas a, b, c e d.
- 4) Dentre as áreas de conhecimento do SWEBOk, temos a área chamada *processo de engenharia de software*. Nela, o foco está nas atividades técnicas e gerenciais dentro do processo. Baseando-se nesta afirmativa, analise as opções abaixo e assinale as que correspondam aos focos da área de processo de engenharia de *software*:
- A) definição do processo, ciclo de vida, avaliação e melhoria do processo, métricas e ferramentas.
- B) Iniciação e definição de escopo, planejamento do projeto, revisão e avaliação.
- C) Gerenciamento de configuração, identificação de configuração e controle de configuração.
- D) Fundamentos de teste, níveis de teste e técnicas de teste.

- E) Fundamentos da manutenção, processos de manutenção e níveis de teste.
- 5) O guia SWEBOK nasceu com o propósito de descrever um corpo de conhecimento não engessado para nortear as atividades de desenvolvimento de *software*.

Para compor o guia, foram pensados objetivos principais. Analise as opções a seguir e assinale o objetivo que foi construído e apoiado por diversos profissionais da área e de vários países.

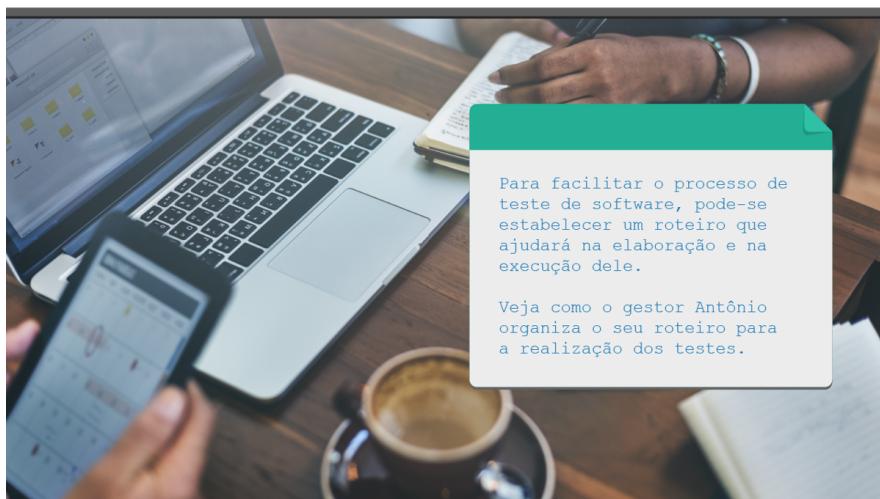
- A) Promover uma visão consistente da engenharia de *software* no âmbito mundial.
- B) Fornecer uma visão geral sobre o conjunto de conhecimentos em gerenciamento de projetos.
- C) Disseminar as melhores práticas de gerenciamento de projetos em todo o mundo.
- D) Atingir uma meta previamente estipulada.
- E) Determinar e aprimorar um modelo para melhoria e avaliação dos processos e serviços.

Na prática

Teste de *software* é uma atividade executada para avaliar a qualidade do produto, buscando identificar os defeitos e problemas existentes (SWEBOK, 2014). Em um processo de testes de *software*, é comum a utilização de *checklist* para auxiliar nos testes e executar os trabalhos de forma mais organizada.

Veja no Na Prática alguns itens que podem ser utilizados para compor um *checklist* para efetuar testes de *software*.

TESTE DE SOFTWARE



Para facilitar o processo de teste de software, pode-se estabelecer um roteiro que ajudará na elaboração e na execução dele.

Veja como o gestor Antônio organiza o seu roteiro para a realização dos testes.



Aponte a câmera para o código e accese o link do conteúdo ou clique no código para acessar.

Roteiro:

Definir itens, requisitos e funcionalidades que serão testados.
Definir o escopo e a abrangência dos testes que serão realizados.
Definir uma abordagem que atenda aos requisitos de qualidade.
Preparar o ambiente de teste.



Para a realização dos testes:

1. Verificar a ortografia das mensagens e dos campos.
2. Verificar o layout do sistema, mantendo o mais próximo possível com o protótipo.
3. Preencher os campos de texto com caracteres especiais e/ou caracteres inválidos.
4. Verificar se os campos estão habilitados ou desabilitados, conforme a especificação.
5. Verificar mensagens como, por exemplo, se ao confirmar a inclusão de cadastro, aparece uma mensagem informando que o item foi cadastrado.
6. Verificar se os campos atualizados foram realmente atualizados.
7. Verificar se na tela de alteração todos os campos foram carregados corretamente.
8. Verificar se ao confirmar a operação de atualização aparece a mensagem informando que o item foi atualizado.
9. Verificar se o registro não é excluído ao clicar em cancelar na mensagem de confirmação de exclusão de registro.
10. Verificar se o registro foi excluído ao clicar em OK na mensagem de confirmação.
11. Realizar uma pesquisa para verificar se a exclusão foi realizada realmente.
12. Verificar se aparece uma mensagem informando que o item foi excluído.



Estes são apenas alguns itens que podem ser inseridos no checklist para a realização dos testes. Aumenta a eficiência manter o resultado deles em um relatório.

Saiba mais

Para ampliar o seu conhecimento a respeito desse assunto, veja abaixo as sugestões do professor:

Qualidade, qualidade de *software* e garantia de qualidade de *software* são as mesmas coisas?

Este artigo mostra as diferenças entre os termos *qualidade*, *qualidade de software* e *garantia da qualidade de software*. Com este artigo, é possível esclarecer a diferença e até mesmo algumas relações entre os três termos.



Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.

IEEE Computer Society

Neste site em inglês, você encontra mais informações referentes ao IEEE e o Guia SWEBOK.



Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.

SWEBOK | Qualidade de *software*

Neste vídeo, você verá sobre o que é o SWEBOK, área de conhecimento e objetivos, além disso, verá um pouco mais sobre qualidade.



Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.