

# ENGENHARIA DE SOFTWARE

Adriana de Souza  
Vettorazzo

# Modelos de qualidade de software

## Objetivos de aprendizagem

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

- Descrever o gerenciamento da qualidade de *software*.
- Identificar os atributos e princípios da qualidade de *software*.
- Indicar normas e modelos de qualidade de *software*.

## Introdução

A qualidade de *software* divide-se em três atividades principais: **garantia da qualidade**, para estabelecer procedimentos e padrões de desenvolvimento que resultam em um *software* de qualidade; **planejamento da qualidade**, processo de desenvolvimento de um plano de qualidade para um determinado processo; e **controle da qualidade**, que garante que o processo especificado seja seguido.

Neste capítulo, você vai estudar sobre modelos de qualidade de *software*, aprendendo a descrever seu gerenciamento e identificando seus atributos, princípios, normas e modelos.

## Gerenciamento da qualidade de *software*

Segundo Sommerville (2011), existem três níveis de preocupação para que a qualidade de um produto de *software* seja alcançada.

- **Preocupação no nível organizacional:** o gerenciamento se preocupa em estabelecer um quadro de processos organizacionais e padrões que resultarão em um *software* de alta qualidade.
- **Preocupação no nível do projeto 1:** o gerenciamento envolve processos quanto à aplicação de processos de qualidade específicos e à

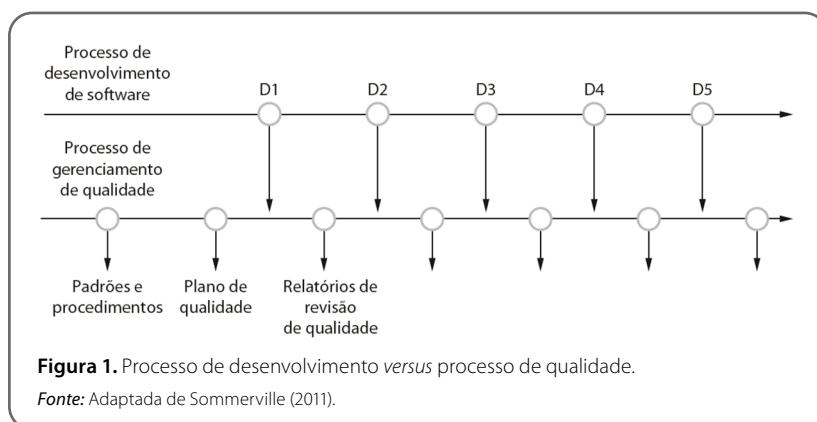
verificação sobre o seu seguimento conforme o planejado, garantindo que tudo esteja correto.

- **Preocupação no nível do projeto 2:** o gerenciamento da qualidade preocupa-se com o desenvolvimento de um plano de qualidade para o projeto. Nesse plano, devem estar descritas as metas de qualidade e a definição de processos e padrões que serão utilizados.

Ainda segundo Sommerville (2011), o gerenciamento de qualidade em sistemas de grande porte pode ser estruturado em três atividades principais:

- fornecer uma verificação independente a respeito do processo de desenvolvimento de *software*;
- verificar as entregas do projeto para garantir que sejam consistentes com os objetivos e padrões organizacionais;
- a equipe de desenvolvimento deve ser diferente da equipe da qualidade, conseguindo, assim, ter uma visão mais objetiva do *software* e gerando relatórios sobre a qualidade sem a influência das questões de desenvolvimento.

A Figura 1 mostra a relação entre as atividades de gerenciamento de qualidade e de desenvolvimento do *software*.



**Figura 1.** Processo de desenvolvimento versus processo de qualidade.

*Fonte:* Adaptada de Sommerville (2011).

## Planejamento da qualidade de software

Planejar a qualidade significa traçar um plano de qualidade para um projeto a partir de padrões apropriados para o produto e o processo. Esse plano definirá

os atributos mais significativos e também a forma como eles serão avaliados, estabelecendo, ainda, os padrões da qualidade que devem ser aplicados e definindo novos padrões a serem utilizados, se necessário.

Um bom plano de qualidade deve apresentar a seguinte estrutura:

- introdução ao produto;
- planos de produto;
- descrições de processo;
- metas de qualidade;
- riscos e gerenciamento de riscos.

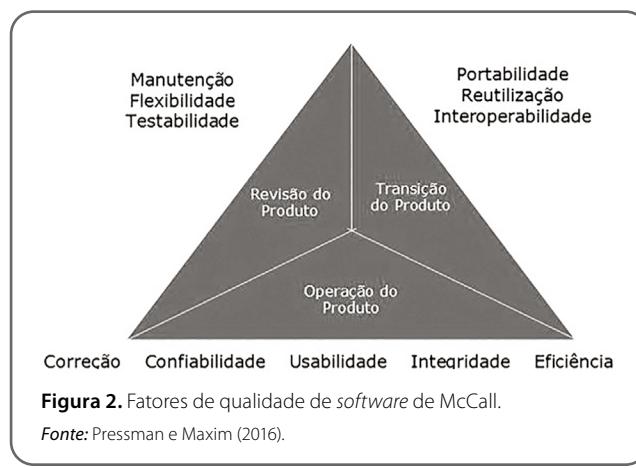
Esse documento deve conter todas as informações necessárias para que sejam realizadas as atividades da qualidade do projeto, além de ser curto e sucinto, de modo a facilitar a leitura da equipe.

Precisa, além disso, registrar o progresso e dar apoio à continuidade do desenvolvimento mesmo que haja mudanças na equipe de desenvolvimento.

Em sistemas menores, o gerenciamento de qualidade necessita de uma documentação menor e deve concentrar-se em estabelecer uma cultura de qualidade.

## Qualidade de software

Para garantir a qualidade, o produto de *software* deve corresponder às expectativas e às especificações. Para isso, McCall, Richards e Walters (1977 apud PRESSMAN; MAXIM, 2016) criaram uma proposta com fatores que afetam a qualidade (Figura 2).



A partir desses fatores, podemos compor os atributos de qualidade de *software* (Quadro 1), que se dividem em três grupos: segurança, compreensibilidade e portabilidade, segundo Sommerville (2011).

**Quadro 1.** Atributos de qualidade de *software*

Segurança	Portabilidade	Compreensibilidade
Proteção	Testabilidade	Usabilidade
Confiabilidade	Adaptabilidade	Reusabilidade
Resiliência	Modularidade	Eficiência
Robustez	Complexidade	Capacidade de aprendizado

*Fonte:* Adaptado de Sommerville (2011).

- **Segurança:** é a medida da capacidade do sistema em resistir a tentativas não autorizadas de utilização ou negação de serviço ao mesmo tempo que presta serviços aos legítimos utilizadores. Para isso, o sistema precisa garantir proteção, confiabilidade, resiliência e robustez.
- **Compreensibilidade:** abrange a capacidade do produto de ser compreendido, aprendido, operado e ter interface atraente ao usuário.
- **Portabilidade:** é a garantia de que o *software* irá desempenhar as funções a que se destina, sendo fácil de se adaptar a diferentes ambientes sem a necessidade de ações adicionais.

Embora tenhamos o mapeamento de todos esses atributos, nem todos os sistemas poderão ser otimizados e utilizados em sua totalidade. Assim, poderão ser utilizados os atributos que se considere mais importante para o *software* que está sendo desenvolvido no momento.

No plano de *software*, pode-se incluir os processos de avaliação da qualidade, o que se transforma em uma forma acordada para avaliar a qualidade.

## Padrões, normas e modelos de qualidade de software

Segundo Sommerville (2011), há uma estreita ligação entre padrões de produto e de processo. Os de produtos aplicam-se à saída do processo de *software*, e os padrões de processo incluem atividades específicas de processos que asseguram que os padrões sejam seguidos.

Os padrões definem os atributos necessários de um produto ou processo e desempenham um papel importante no gerenciamento de qualidade. Assim, enquanto os padrões de produto definem as características que os componentes de *software* devem exibir, os padrões de processo definem como o processo de *software* deve ser seguido. Veja, no Quadro 2, alguns padrões que podem ser utilizados.

**Quadro 2.** Padrões de produto e processo

Padrões de produto	Padrões de processo
Formulário de revisão de projeto	Condução de revisão de projeto
Estrutura de documento de requisitos	Apresentação do novo código para a construção de sistema
Formato de cabeçalho de método	Processo de versão e <i>release</i>
Estilo de programação Java	Processo de aprovação de plano de projeto
Formato de plano de projeto	Processo de controle de mudança
Formulário de solicitação de mudança	Processo de registro de teste

*Fonte:* Adaptado de Sommerville (2011).

Os padrões podem mudar com o tempo por necessidade ou pelo surgimento de novos. Assim, para evitar que membros da equipe os desconsiderem por

achar que são burocráticos ou até mesmo irrelevantes para o desenvolvimento de atividades técnicas, alguns passos podem ser seguidos:

1. envolver a equipe de *software* para selecionar os padrões que serão adotados para o produto;
2. fazer a revisão dos padrões regularmente para refletir as mudanças de tecnologia;
3. utilizar ferramentas de *software* que apoiem a utilização dos padrões escolhidos.

Existe um número significativo de normas técnicas. Veja, a seguir, no Quadro 3, algumas normas internacionais e modelos de qualidade de *software*.

**Quadro 3.** Normas internacionais e modelos de qualidade de *software*

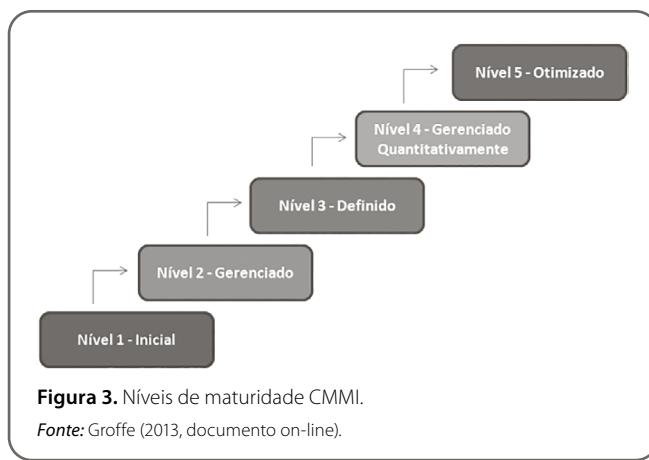
<b>Produto</b>		<b>Processo</b>	
ISO 9126	Norma para qualidade de produtos de <i>software</i>	ISO 12207	Processos de ciclo de vida do <i>software</i>
ISO 14598	Guias para avaliação de produtos de <i>software</i>	ISO 90003:2004	Diretrizes para aplicação da norma ISO 9001 ao desenvolvimento, fornecimento e manutenção de <i>software</i>
		ISO 15504 (SPICE)	Projeto da ISO/IEC para avaliação dos processos de desenvolvimento de <i>software</i>
		CMMI	<i>Capability Maturity Model Integrated</i> . Modelo do SEI que estende o CMM para avaliação de processos de <i>software</i>
		MPS-BR	Modelo Brasileiro de qualidade de processo de <i>software</i> baseado nas normas ISO 12207 e 15504 e no modelo CMMI

## Modelo CMMI

O modelo CMMI (*Capability Maturity Model Integration*) foi criado pelo SEI *Software Engineering Institute* e tem foco na capacidade de maturidade de processos de *software*, ou seja, serve de guia para a melhoria de processos.

O CMMI apresenta cinco níveis de maturidade, que estabelecem patamares de evolução de processos, caracterizando estágios de melhoria da implementação de processos (Figura 3).

- **Nível 1 – inicial:** os processos são caóticos e a organização não apresenta ambiente estável de desenvolvimento. Se existem padrões, eles não são seguidos, e os projetos, geralmente, apresentam problemas de execução dos projetos no que diz respeito a prazos e custos.
- **Nível 2 – gerenciado:** os projetos possuem requisitos gerenciados e processos planejados. Os requisitos, processos e serviços são gerenciados e há uma preocupação em seguir o que está determinado nos planos.
- **Nível 3 – definido:** os processos são caracterizados e bem entendidos. Existe a padronização dos processos, o que possibilita que os produtos gerados sejam consistentes. Nesse estágio, os procedimentos são padronizados e devem prever a aplicação em diferentes projetos.
- **Nível 4 – gerenciamento da qualidade:** os processos selecionados contribuem para o desempenho geral dos demais processos. Eles são controlados com base em estatísticas e técnicas quantitativas.
- **Nível 5 – otimizado:** os processos são continuamente avaliados e melhorados. Os objetivos quantitativos são estabelecidos e revisados com frequência.



Alguns benefícios da implantação do CMMI são:

- maior confiabilidade no que se refere ao cumprimento de prazos e custos que foram acordados;
- o acompanhamento das atividades relativas à produção de *software* aumenta consideravelmente;
- aumento da qualidade dos *softwares* criados devido aos processos bem definidos.
- já que os processos são bem definidos, a dependência por profissional com elevado conhecimento técnico diminui;
- a melhoria contínua é uma busca diária.

## Modelo MPS–BR

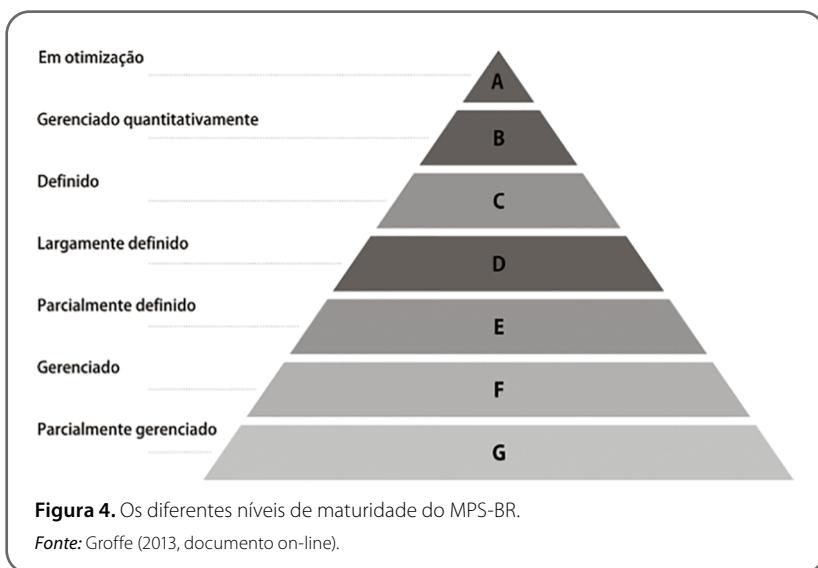
O modelo de Melhoria do Processo de Software Brasileiro (MPS–BR) é uma metodologia voltada à área de desenvolvimento de sistemas.

A exemplo do modelo CMMI, o MPS-BR também apresenta níveis de maturidade.

- **Nível A – em otimização:** um processo nesse nível é otimizado por meio da realização de mudanças e adaptações de forma ordenada, com foco na efetividade para atender mudanças nos objetivos do negócio.
- **Nível B – gerenciado quantitativamente:** a organização passa a ter uma visão quantitativa quanto ao desempenho dos seus processos.
- **Nível C – definido:** são implementados três novos processos com a mesma capacidade dos processos já implantados:
  - análise e decisão e resolução (ADR);
  - desenvolvimento para a reutilização (DRU);
  - gerência de riscos (GRI).
- **Nível D – largamente definido:** estar nesse nível implica apenas a definição e a implementação de cinco novos processos com a mesma capacidade dos processos já implantados:
  - desenvolvimento de requisitos (DRE);
  - integração do produto (ITP);
  - projeto e construção do produto (PCP);
  - validação (VAL);
  - verificação (VER).

- **Nível E – parcialmente definido:** o foco principal é a padronização dos processos da organização, que devem ser definidos a partir dos processos e das melhores práticas já existentes.
- **Nível F – gerenciado:** existe o apoio à gestão do projeto referente à garantia da qualidade e medição. Esses processos possibilitam maior visibilidade de como os artefatos são produzidos.
- **Nível G – parcialmente gerenciado:** a implementação deve ser executada com cautela devido à mudança de cultura organizacional e à definição do conceito acerca do que é projeto para a organização.

Veja, na Figura 4, os níveis de maturidade do MPS-BR.



Alguns benefícios da implantação do MPS-BR são:

- o MPS-BR é mais adequado à realidade brasileira e mais acessível do que o modelo CMMI (Figura 5);
- maior número de níveis: tem sete níveis de maturidade e a implantação é mais gradual e adequada a pequenas e médias empresas;
- compatibilidade com CMMI, facilitando a obtenção do certificado;
- as empresas são avaliadas a cada 2 anos para manter o certificado ou para evoluir ao próximo nível.

		CORRELAÇÃO CMMI E MPS.BR	
		CMMI	MPS.BR
5	Análise Casual e Resolução – CAR Inovação e Melhoria Organizacional - OID	A	Análise de Causas de Problemas e Resolução
4	Desempenho do Proc. Org. – OPP Gerência Quantitativa de Projeto - QPM	B	Gerência Quantitativa do Projeto
3	Foco no Processo da Organização – OPF Definição do Proc. da Organização – OPD Treinamento Organizacional – OT Gerência Integrada de Projeto – IPM Gerência de Risco – RSKM Desenvolvimento de Requisitos – RD Solução Técnica – TS Integração do Produto – PI Verificação – VER Validação – VAL Análise de Decisão e Resolução - DAR	C	Análise de Decisão e Resolução Gerência de Riscos Desenvolvimento de Reutilização
		D	Desenvolvimento de Riscos Integração do Produto Projeto e Construção do Produto Verificação Validação
		E	Gerência de Recursos Humanos Avaliação e Melhoria do Proc. Org. Definição do Proc. Organizacional Gerência de Reutilização
2	Gerência de Requisitos – REQM Planejamento de Projeto – PP Acompanhamento e Contr. de Proj. – PMC Ger. de Acordo com Fornecedores – SAM Gar. de Qual. de Proc. e Produto – PPCQA Gerência de Configuração – CM Medição e Análise - MA	F	Medição Gerência de Configuração Aquisição Garantia da Qualidade
		G	Gerência de Requisitos Gerência de Projetos

**Figura 5.** Principais semelhanças entre os modelos CMMI e MPS-BR.

**Fonte:** Franciscani e Pestili (2012, documento on-line).



### Fique atento

O plano de qualidade deve estabelecer as qualidades desejadas para o *software* e, também, descrever como elas serão avaliadas.



## Referências

FRANCISCANI, J. F.; PESTILI, L. C. *CMMI e MPS.BR*: um estudo comparativo. 2012. Disponível em: <<http://www.unicerp.edu.br/images/revistascientificas/3%20-%20CMMI%20e%20MPS.BR%20Um%20Estudo%20Comparativo1.pdf>>. Acesso em: 01 nov. 2018.

GROFFE, R. J. *Maturidade no desenvolvimento de software: CMMI e MPS-BR*. 2013. Disponível em: <<https://www.devmedia.com.br/maturidade-no-desenvolvimento-de-software-cmmi-e-mps-br/27010>>. Acesso em: 01 nov. 2018.

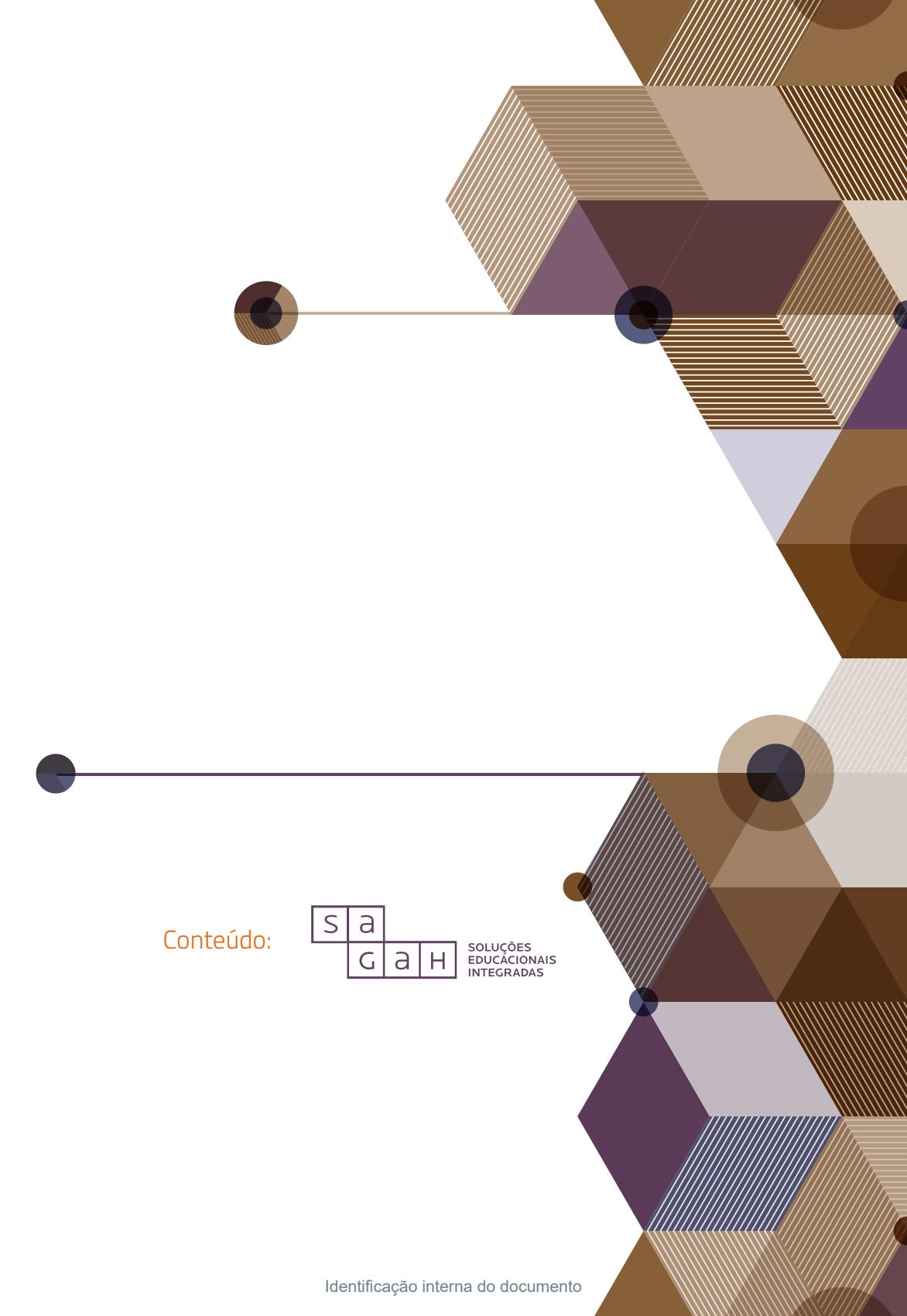
PRESSMAN, R. S.; MAXIM, B. R. *Engenharia de software*: uma abordagem profissional. 8. ed. Porto Alegre: Bookman, 2016.

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

## Leitura recomendada

STEFFEN, J. B. *O que são essas tais de metodologias ágeis?* 2012. Disponível em: <[https://www.ibm.com/developerworks/community/blogs/rationalbrasil/entry/mas\\_o\\_que\\_s\\_c3\\_a3o\\_essas\\_tais\\_de\\_metodologias\\_\\_c3\\_a1geis?lang=en](https://www.ibm.com/developerworks/community/blogs/rationalbrasil/entry/mas_o_que_s_c3_a3o_essas_tais_de_metodologias__c3_a1geis?lang=en)>. Acesso em: 01 nov. 2018.

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