PERSONAL SOFTWARE PROCESS: UMA VISÃO GERAL SOBRE O PROCESSO E O SEU IMPACTO NA INDÚSTRIA DE SOFTWARE

PERSONAL SOFTWARE PROCESS: AN OVERVIEW OF THE PROCESS AND ITS IMPACT ON SOFTWARE INDUSTRY

Antonio Marcos Neves Esteca[*]
Jorge Paulo Fernandes[**]
Hil Marques Silva Júnior[***]
Raquel da Anunciação[****]

RESUMO

O crescente uso de sistemas de informação nas organizações acarreta a busca por qualidade, com o objetivo de promover a agilidade no acesso a informações confiáveis. Para a obtenção de software de qualidade, é essencial o emprego de um processo de desenvolvimento de software pautado em boas práticas. Nesse contexto, esse trabalho apresenta um processo para capacitação individual e monitoramento rigoroso dos desenvolvedores de software, denominado *Personal Software Process* - PSP, o qual tem levado a manutenção de maior disciplina e controle sobre todas as fases do desenvolvimento. Apesar de seus benefícios, a implantação do PSP demanda esforço e comprometimento das indústrias de software, uma vez que pode enfrentar barreiras e dificuldades oriundas das mudanças que o processo propõe na rotina de trabalho dos desenvolvedores de software.

Palavras-chave: PSP. Qualidade de Software. Processo de Software.

ABSTRACT

The growing use of software systems in business has resulted in a search for quality so as to increase access agility to reliable information. To obtain quality software, it is essential to follow a software development process based on good practices. In this context, this paper presents a process of individual qualification and strict monitoring of team members, called Personal Software Process - PSP, which has led to maintenance of greater discipline and control throughout all development phases. Despite its benefits, the adoption of PSP demands effort and commitment of software industries, since it may face barriers and difficulties arising from changes proposed by the process in the work routine of software developers.

Keywords: PSP. Software Quality. Software Process.

[*] Docente da UNIESP (Ribeirão Preto-SP) e orientador desta pesquisa. am.esteca@sjrp.unesp.br
[**]Graduado em Sistemas de Informação pela UNIESP (Ribeirão Preto-SP). jorgepaulofernandes@yahoo.com.br
[***]Graduado em Sistemas de Informação pela UNIESP (Ribeirão Preto-SP). hillqra@yahoo.com.br;
[****]Graduado em Sistemas de Informação pela UNIESP (Ribeirão Preto-SP). raquel_anunciacao@yahoo.com.br.

Introdução

Na visão popular, qualidade é uma palavra com um significado muito subjetivo. O que pode ser de ótima qualidade para uma pessoa pode não ser para outra. No entanto, no contexto de software, qualidade é um conceito bem definido e de grande importância às indústrias de software.

Segundo Pressman (2009), qualidade de software consiste na conformidade com requisitos funcionais e não-funcionais explicitamente declarados, normas de desenvolvimento documentadas e características implícitas esperadas em todo o software desenvolvido profissionalmente.

Nas últimas décadas, a preocupação com a qualidade de software tem crescido abruptamente, pois organizações de todos os setores estão se tornando dependentes dos sistemas de informação para a execução de tarefas e gestão dos seus negócios (SILVA, 2006).

Como em outras áreas, a qualidade do produto de software resulta do processo empregado durante seu desenvolvimento, o que tem levado as indústrias de software a valorizarem os profissionais capacitados para propor, otimizar e implantar processos.

De acordo com Paula Filho (2009), processo pode ser entendido como uma receita a ser seguida para a produção de algo. Já em uma visão mais técnica, processo pode ser definido como um conjunto de atividades inter-relacionadas realizadas para obter um conjunto específico de produtos, resultados ou serviços (PMI, 2013).

Visando a melhoria da qualidade do processo de software, surge o *Personal Software Process* ou Processo Pessoal de Software, também conhecido pela sigla PSP, que pode ser definido como um processo de desenvolvimento de software desenvolvido para ser utilizado por desenvolvedores na construção de sistemas de software. O PSP foi criado em 1989 pelo engenheiro de software norte-americano Watts Humphrey (HUMPHREY, 1989) e posteriormente descrito em detalhes no livro "*A Discipline for Software Engineering*" (Uma disciplina para Engenharia de Software), de 1995 (HUMPHREY, 1995). Humphrey faleceu em 2010 e é considerado o pai da Qualidade de Software.

Em termos gerais, o PSP consiste em um conjunto de formulários, diretrizes e procedimentos estatísticos complexos que, quando empregados corretamente, proveem dados importantes para que os desenvolvedores de software conheçam melhor seus

compromissos e dificuldades, tornando suas rotinas de trabalho mais previsíveis e eficientes (HUMPHREY, 1995).

1 Objetivo

O objetivo deste artigo é mostrar, em caráter introdutório, as metas do PSP, sua organização, os benefícios que sua implantação pode trazer, bem como os custos para sua implantação. O texto foi elaborado de modo que possa ser entendido não apenas por especialistas em qualidade de software, mas também por estudantes de Tecnologia da Informação (TI) que tenham interesse em conhecer processos ainda pouco praticados no Brasil, e que podem trazer resultados excelentes às organizações, como tem sido experimentado em diversos países. Com isso, espera-se que esse artigo sirva como uma leitura seminal para estudantes e profissionais de TI, que estarão aptos, após essa introdução, a buscar fontes de conteúdo mais avançado sobre o PSP.

2 Metodologia

A metodologia utilizada para criação deste artigo baseou-se no estudo de material bibliográfico relacionado ao tema proposto. Para tanto, inicialmente foram identificados livros, artigos científicos, monografias e dissertações relacionados à qualidade de software e ao PSP. Em seguida, os materiais de melhor qualidade foram selecionados e, a partir daí, estudados detalhadamente.

Na Figura 1 é ilustrado o processo metodológico adotado para o desenvolvimento desse trabalho.

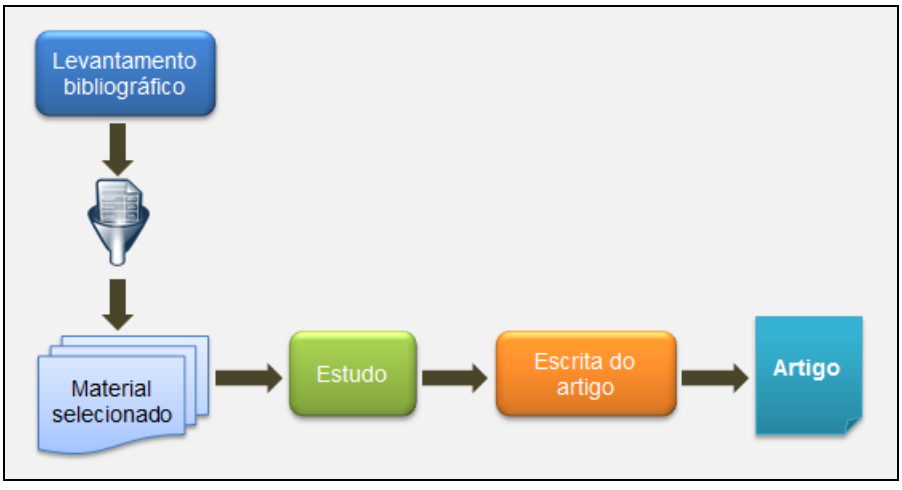

Figura 1 – Processo metodológico

3 O *Personal Software Process*

O PSP propõe que o processo de desenvolvimento de software seja dividido em fases, durante as quais devem ser aplicados seus componentes. Na Figura 2 é ilustrado o fluxo dos componentes do PSP ao longo de suas fases. Conforme é apresentado, os requisitos constituem a entrada inicial do processo, sendo processados desde a fase de planejamento, passando por todo o desenvolvimento do software até chegar ao produto final. Durante todas essas fases do PSP, os processos devem ser guiados pelos s*cripts* e padrões e, por meio de formulários, devem ser realizados registros obtidos com a medição de tempo e defeitos. Ao concluir a construção de um programa, o plano criado na fase de planejamento e os dados reais registrados nos formulários devem ser confrontados, gerando o Resumo de Planejamento.

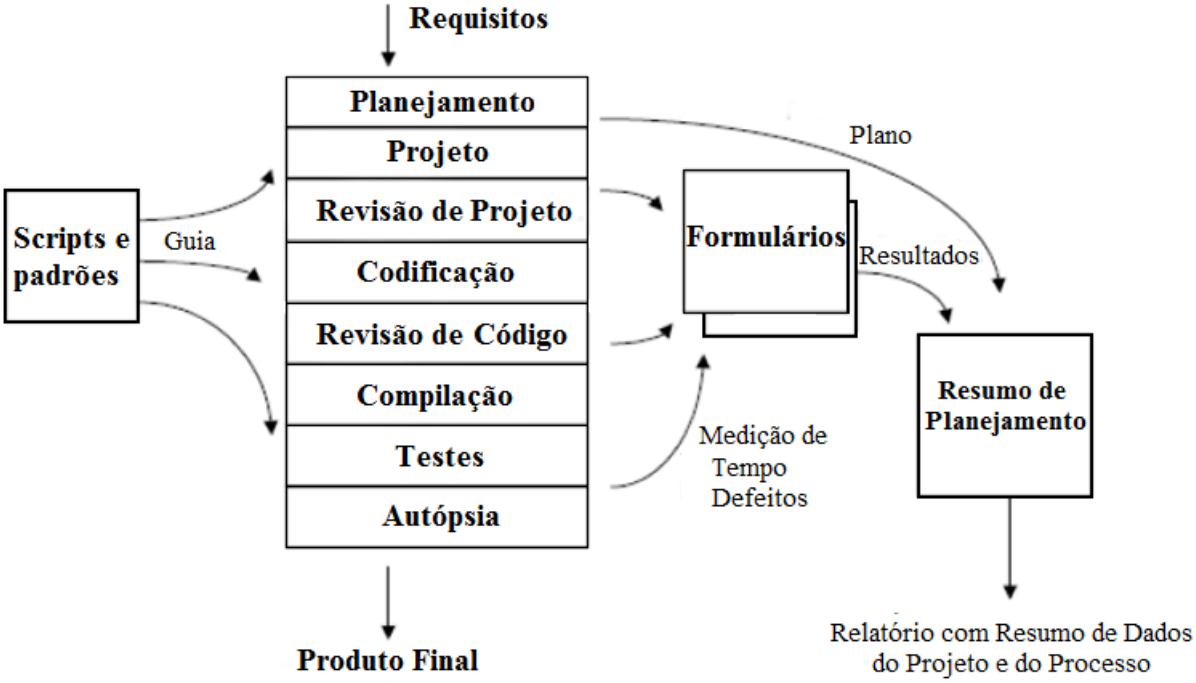

Figura 2 – Fluxo do PSP

Para apoiar o aumento da maturidade dos desenvolvedores de software, o PSP é dividido em níveis, que introduzem processos de forma gradativa à rotina de construção de software.

Nas próximas subseções são apresentadas de forma mais detalhada as fases do PSP, seus componentes e seus níveis.

3.1 Fases do PSP

Com o intuito de estabelecer um processo de software de qualidade, o PSP divide o processo de software em três fases (POMEROY-HUFF et al, 2009):

1. **Planejamento:** será definido o que fazer e como fazer, sendo produzido um plano para direcionar o desenvolvimento do trabalho;

2. **Desenvolvimento:** Esta fase é o momento em que se realiza o trabalho segundo as seguintes etapas:
    - Definição de requisitos – detalhamento das funções que o software deve contemplar e das restrições sob as quais deve operar;
    - Projeto do software – modelagem do software em desenvolvimento;
    - Revisão do projeto – revisão sistemática do projeto elaborado, de modo a corrigir precocemente os defeitos;
    - Codificação do programa – programação do software em desenvolvimento;
    - Revisão do código – revisão técnica do código construído, de modo a encontrar defeitos antes mesmo da compilação, o que garante uma correção menos custosa;
    - Compilação - compilação e execução do software construído, com a correção dos defeitos que forem encontrados;
    - Teste do programa – fase de teste do software desenvolvido.

3. ***Postmortem* ou autópsia:** estabelece comparativos dos dados planejados e reais e documenta todas as ideias para melhoria do processo.

3.2 Os componentes do PSP

O PSP é composto por quatro elementos básicos:

1. ***Scripts* ou Roteiros:** são descrições de fácil entendimento que direcionam o desenvolvedor na realização de um processo, sendo compostos por:
    - Objetivo do processo;
    - Critérios de entrada;

- Diretrizes gerais;
- Passos a serem executados;
- Critérios de qualidade;
- Condições para encerramento do processo.

2. **Formulários:** definem os dados a serem coletados a cada nível do PSP. Exemplos de formulários: *Time Recording Log, Checklists*, etc.

3. **Medições:** quantificam o processo e o produto e habilitam os desenvolvedores a:
    - Desenvolver perfis dos dados de projetos anteriores, os quais podem ser usados como base para planejamentos;
    - Analisar um processo para determinar como melhorá-lo;
    - Determinar a efetividade das modificações do processo;
    - Monitorar a execução do processo;
    - Monitorar a produtividade pessoal.

4. **Padrões:** oferecem definições precisas e consistentes que guiam o trabalho e a coleta e uso dos dados, tornando possível aplicar medições uniformemente em vários projetos.

3.3 Níveis do PSP

O PSP é dividido em sete níveis evolucionários e sua implantação é realizada de forma incremental, adicionando aos níveis já implantados as práticas propostas pelos níveis seguintes. Assim, progressivamente o desenvolvedor aprimora habilidades importantes para a melhoria contínua da qualidade dos sistemas de software produzidos.

Na Figura 3 é ilustrada a estrutura do PSP, sumarizando os elementos incorporados a cada nível.

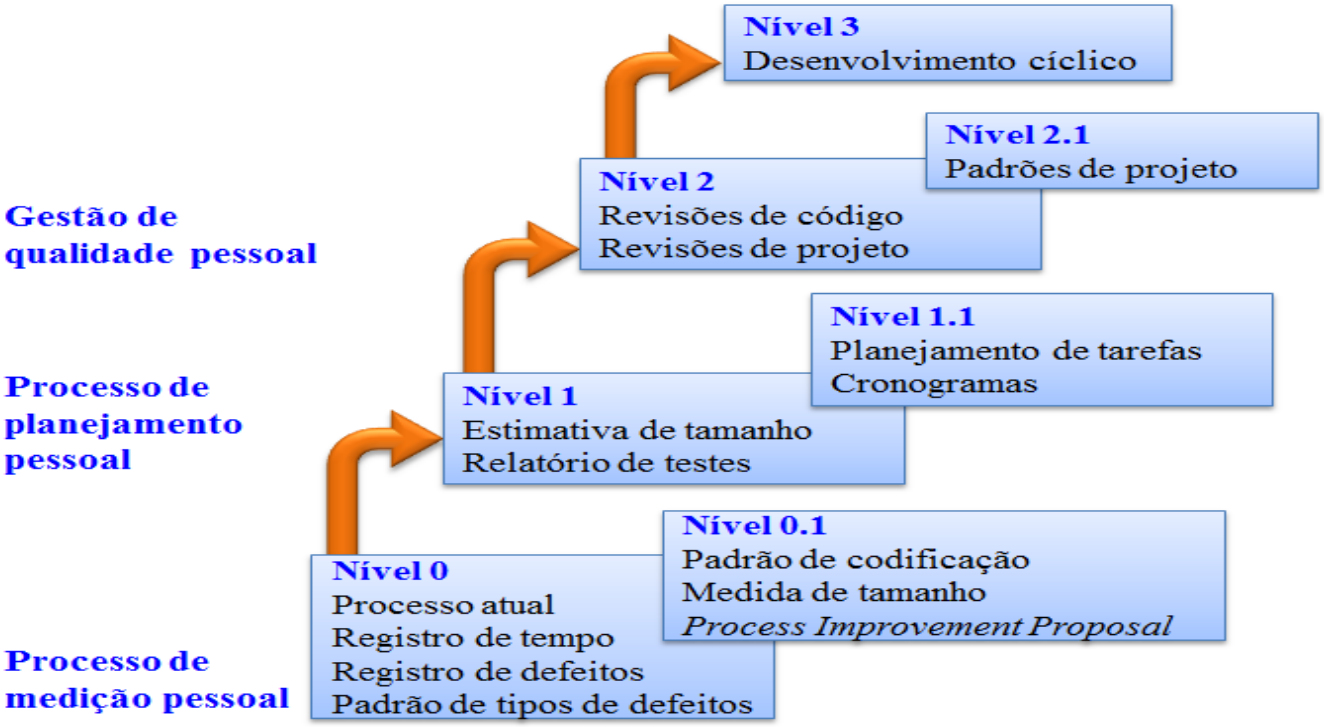

Figura 3 – Níveis do PSP

Conforme é ilustrado na Figura 3, o PSP é dividido em sete níveis:

- **Nível 0** – Também chamado de PSP 0, esse nível introduz as primeiras medições e relatórios padronizados à rotina dos desenvolvedores. Os objetivos do nível 0 do PSP são (HUMPHREY, 2005):

  - usar o processo atual como um processo introdutório;

  - incorporar a medição do tempo gasto em cada atividade dos projetos, por meio do formulário *Time Recording Log;*

  - incorporar a medição dos defeitos encontrados, com o emprego do *Defect Recording Log*, que registra o tempo gasto na correção dos defeitos, além de informações sobre o tipo dos defeitos e as fases em que eles foram injetados e removidos dos programas.

- **Nível 0.1** – Também chamado de PSP 0.1, esse nível é uma evolução do nível 0. Os objetivos do nível 0.1 do PSP são (HUMPHREY, 2005):

  - estimular a criação e o uso de um padrão de codificação;

  - incorporar a medição do tamanho dos programas desenvolvidos (em número de linhas de código);

- incorporar o registro de propostas de melhoria ao processo de software, por meio do formulário *Process Improvement Proposal* (PIP).

- **Nível 1** – Também chamado de PSP 1, esse nível tem como foco as fases de planejamento e desenvolvimento. Os objetivos do nível 1 do PSP são (HUMPHREY, 2005):

  - incorporar um método estatístico avançado para estimar o tamanho e tempo de desenvolvimento de um software. Esse método é denominado *Proxy-Based Estimating* (PROBE);

  - incorporar a elaboração de relatório sobre os testes executados nos programas.

- **Nível 1.1** – Também chamado de PSP 1.1, esse nível é uma evolução do nível 1. Os objetivos do nível 1.1 do PSP são (HUMPHREY, 2005):

  - elaborar cronogramas dos projetos de desenvolvimento de software;

  - empregar técnicas para monitorar o progresso dos projetos, bem como detectar atrasos em relação ao planejado.

- **Nível 2** – Também chamado de PSP 2, esse nível tem como foco o gerenciamento de qualidade pessoal. Os objetivos do nível 2 do PSP são (HUMPHREY, 2005):

  - introduzir revisões de projeto e de código baseadas em *checklists* pré-definidos para direcionar as revisões;

  - gerar diversos indicadores de qualidade sobre o processo de software, o que é feito com base nos diversos dados coletados por meio das medições realizadas nos níveis anteriores;

  - estimar o número de defeitos que provavelmente serão gerados, de modo a tentar reduzi-los com o passar do tempo.

- **Nível 2.1** – Também chamado de PSP 2.1, esse nível é uma evolução do nível 2. Os objetivos do nível 2.1 do PSP são (HUMPHREY, 2005):

  - reduzir o número de defeitos dos programas desenvolvidos;

  - estimular o uso de modelos consagrados para projetar e documentar os sistemas desenvolvidos.

- **Nível 3** – A abordagem nesta fase busca subdividir o programa em partes menores (módulos), para que sejam recepcionados novamente pelas fases anteriores garantindo a qualidade fase a fase, pois se uma iteração anterior contiver falhas, importantes benefícios serão perdidos.

4 Benefícios do PSP

Estudos realizados em indústrias de software revelam que, através de sua estrutura robusta e bem definida, o PSP oferece diversos benefícios pessoais e/ou organizacionais, cujos principais são (JOHNSON et al, 2003; RAMINGWONG; RAMINGWONG, 2012):

- aumento da produtividade dos desenvolvedores;
- diminuição do número de defeitos gerados;
- melhor planejamento e controle do cronograma de projetos de software através de estimativas, acompanhamento e medições;
- melhoria da qualidade dos produtos de software;
- diminuição do tempo de realização de testes nos produtos de software e consequente redução do tempo e custo total de desenvolvimento, o que pode representar uma vantagem competitiva às indústrias;
- aperfeiçoamento do processo de desenvolvimento de software, a partir de dados coletados pelo formulário PIP;
- atendimento, pelo menos parcial, de 12 das 18 áreas chave de processo do *Capability Maturity Model* (CMM).

5 Os custos do PSP

O PSP entra em uma empresa com um papel difícil de administrar, pois provoca mudanças de conceitos que podem perturbar o ego dos profissionais envolvidos, ora causando uma impressão de que eles estavam "deixando a desejar" com sua rotina pré-estabelecida, ora aumentando as expectativas de quem é recém-chegado no negócio.

O principal argumento para adoção do PSP em uma empresa é a busca pela qualidade. No entanto, para que o PSP consiga trazer seus benefícios, é necessário implantar diversas práticas que, muitas vezes, podem ser vistas como ameaça pelos

colaboradores, uma vez que eles passam a ser monitorados quanto à produtividade, qualidade e eficiência. Diante disso, é necessário que os gestores saibam introduzir o PSP às suas equipes como uma ferramenta aliada a todos, e não como um controle excessivo do ambiente de produção. Cabe aqui ressaltar que o PSP é um processo de automelhoria, cujos benefícios estão pautados no desenvolvimento da maturidade de cada indivíduo, e não na imposição de pressão sobre eles.

Além da resistência que o PSP pode enfrentar, outra barreira é a sua curva de aprendizagem. A maioria dos cursos sobre o PSP leva em torno de quatro meses e um total de cento e trinta horas aproximadamente, o que pode ser considerado muito demorado por algumas organizações (ESTECA, 2013).

Por fim, o sucesso do PSP a médio e longo prazo depende da real adoção de diversas práticas, que certamente demandarão sempre algum tempo dos desenvolvedores, especialmente no início.

Diante desse cenário, é possível concluir que embora o PSP possa trazer diversos benefícios às indústrias de software, é de suma importância que haja dedicação e comprometimento com as práticas propostas por esse processo.

Considerações Finais

Este artigo apresentou o PSP, um processo de software que direciona o trabalho de cada um dos envolvidos no desenvolvimento de software, com o intuito de promover o autoconhecimento e a melhoria contínua de cada um, tanto em relação à produtividade como em relação à qualidade e eficiência.

Ao longo do artigo, foi apresentada a estrutura geral do PSP, as fases em que ele é organizado e os níveis de maturidade propostos. Conforme apresentado, o PSP parte do processo que o desenvolvedor já aplica em sua rotina de trabalho e propõe a inclusão de novos processos de forma gradativa para promover a melhoria da maturidade dos envolvidos.

O artigo ainda apresentou que, a partir de sua robusta estrutura, o PSP pode trazer diversos benefícios pessoais e organizacionais, permitindo maior previsibilidade do trabalho, melhorando o nível de acerto dos planejamentos e conduzindo ao desenvolvimento de software de melhor qualidade e em menor tempo.

Por fim, o artigo apresentou que apesar dos benefícios que pode trazer, a implantação do PSP pode enfrentar uma série de barreiras, que vão desde o tempo para

aprendizagem do processo até à resistência de colaboradores e gestores de uma indústria de software.

A partir dessa leitura, é possível concluir que o PSP pode trazer importantes benefícios às indústrias de software, mas sua implantação demanda comprometimento e esforço, especialmente na fase inicial.

Espera-se que esse artigo promova uma visão introdutória sobre o PSP capaz de motivar os leitores a se aprofundarem nesse processo, que certamente é um importante diferencial para o currículo de profissionais de TI, sendo prática comum em países desenvolvidos, mas ainda pouco explorada no Brasil (ESTECA, 2013). Além disso, artigos futuros poderão abordar esse tema de forma mais profunda, desenvolvendo até mesmo ferramentas e métodos de apoio ao emprego do PSP.

Referências

ESTECA, A. M. N. **Apoio à maturidade pessoal visando a melhoria dos projetos de software.** 2013. 137 p. Dissertação (Mestrado em Ciência da Computação) – Instituto de Biociências, Letras e Ciências Exatas, Universidade Estadual Paulista, São José do Rio Preto, 2013.

HUMPHREY, W. S. **A discipline for Software Engineering:** the complete PSP book. Reading: Addison-Wesley Publishers, 1995.

_____. Modeling Implications of the Personal Software Process. IN: TERNATIONAL SOFTWARE PROCESS WORKSHOP, 5th, **Proceedings…** Kennebunkport, Estados Unidos, out. 1989.

_____. **PSP:** a self-improvement process for Software Engineers. Reading: Addison-Wesley Publishers, 2005.

JOHNSON, P. M. et al. Beyond the Personal Software Process: metrics collection and analysis for the differently disciplined. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, 25th, **Proceedings…**, Portland, Estados Unidos, maio de 2003.

PAULA FILHO, W. P. **Engenharia de software:** fundamentos, métodos e padrões. 3. ed. Rio de Janeiro: LTC, 2009.

PMI Project Management Institute. **Guide of project management body of knowledge**. 5. ed. Newtown Square: Project Management Institute – PMI, 2013.

POMEROY-HUFF, M. et al. **The Personal Software Process (PSP) body of knowledge.** Technical Report, CMU/SEI-2009-SR-018, v.2.0, ago. 2009.

PRESSMAN, R. **Software Engineering:** a Practitioner's Approach. 7th ed. McGraw-Hill, 2009.

RAMINGWONG, S.; RAMINGWONG, L. Implementing a Personal Software Process (PSPSM) course: a case study. **Journal of Software Engineering and Applications**, v. 5, n. 8, p. 639-644, ago, 2012.

SILVA, R. A. C. **PSP e métodos ágeis na melhoria da qualidade em produção de software: um estudo de caso.** 2006. 128 p. Dissertação (Mestrado em Ciência da Computação) – Centro de Ciências Exatas e Tecnológicas, Universidade Federal de Viçosa, Viçosa-MG, 2006.