Seção 2

**O processo de software**

O cotidiano de um **Analista de Sistemas** é bem agitado, são muitas tarefas que devem ser executadas: visita ao cliente, orientação aos programadores, pesquisas e, claro, muito planejamento.

**Desafio:**

Investigar o **Processo de software** da empresa e propor melhorias

Alguns clientes têm relatado um incômodo, quanto ao tempo de entrega do produto, o software.

Para isso, deverá se concentrar em responder os seguintes questionamentos:

**Quais as atividades principais de um Processo de Software?**

**Como você poderia melhorar o Processo de construção de um software?**

Um **Processo de Software**, conforme destaca Sommerville (2011), é um **Conjunto de Atividades e Resultados** que estão relacionados, que levam à produção e ao resultado de um software desenvolvido.

Pressman (2016) enfatiza que na Engenharia de Softwareum **Processo** não é uma deter­minação rigorosa sobre como ele deve ser desenvolvido, ao contrário, **é uma abordagem adaptável** que possibilita à equipe de desenvolvimento escolher os processos que melhor se enquadram na filosofia da empresa (de desenvolvimento) com o **foco** na **Qualidade do Produto**, **no Prazo de Entrega e na Redução de Custos.**

Sommerville (2011) afirma que a **abordagem sistemática** usada pela **Engenharia de Software** para **Produção de software** é chamada **Processo de Software.**

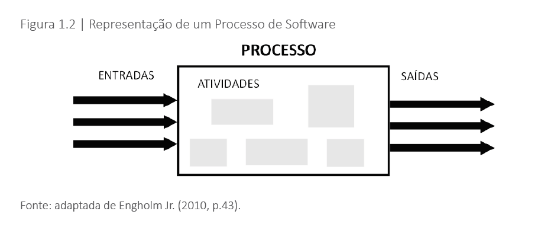
Um **Processo de Software** pode conter **diversas atividades** que normalmente são:

**-Especificação; -Projeto;-Implementação;-Validação;-Manutenção;-Evolução;**

Engholm Jr. (2010) relaciona os seguintes A**spectos** sobre o **Processo de Software**:

1. **-Cria uma Padronização** na forma de gerar os serviços e produtos.
2. **-Permite que sejam** repetidos os serviços e produtos, **reutilizando** partes já produzidas e padronizadas.
3. **-Retém o conhecimento na empresa**, permitido que novos integrantes possam dar continuidade aos processos definidos previamente.
4. **-**Serve para **definir e guiar** as atividades de um **Projeto de Software.**
5. -Permite a **especificação de todo o processo** para o desenvolvimento do software (ou partes dele).
6. **-Determina as tarefas** que deverão ser executadas para a equipe e **de forma individual**.
7. **-Reduz riscos**, tornando os resultados mais previsíveis.
8. **-Proporciona visões comuns** para a equipe de desenvolvimento, facili­tando a comunicação.
9. **-Pode ser** **utilizado** **como um *template***, sendo utilizado em outros projetos, permitindo agilidade em novos projetos de software.
10. **Na definição dos Processos de Software**, segundo Engholm Jr. (2010), são definidas uma série de **parâmetros**:
11. (i) **Evento** que determina o **início** do processo;
12. (ii) **Matriz** de responsabilidades;
13. (iii) **Atividades** que serão execu­tadas e suas sequencialidades;
14. (iv) **Entradas** e **saídas** de cada atividade;
15. (v) **Regras** e **políticas** a serem aplicadas na realização das atividades;
16. (vi) **Infra­estrutura** necessária;
17. (vii) **Resultado** gerado na execução de cada processo.

A Figura 1.2 demonstra graficamente que, em cada **Processo de Software**, podem ocorrer diversas atividades (sequenciais ou em paralelo).



Cada atividade metodológica é **composta** por um **conjunto de ações** de **engenharia de software**.

Cada ação é definida por um **conjunto de tarefas**, o qual identifica as tarefas de trabalho a ser completadas, os artefatos de software que serão produzidos, os fatores de garantia da qualidade que serão exigidos e os macros utilizados para indicar progresso. (PRESSMAN, p. 31, 2016) 

O **Processo de Software** adotado em uma empresa pode ser completa­mente diferente de outra empresa, cada qual procura encontrar e estabelecer atividades que visam aumentar **a qualidade e baixar o custo de produção do software produzido.**

Independente do **modelo de Processos de Software** adotado pela empresa de desenvolvimento, todos utilizam uma **Estrutura de Processo Genérico de Software,** com atividades pré-estabelecidas.

**As atividades** de um determinado **Processo de Software** constituem um **conjunto mínimo** para se obter um produto de software (o software finalizado e entregue ao cliente);

Em um **Processo Genérico de Software**, os processos podem ser diferentes, mas podemos identificar **quatro** atividades fundamen­tais em toda a **produção de software**, conforme Sommerville (2011):

1. **Especificação de Software**: Definição do que será desenvolvido, suas restrições e funcionalidades;
2. **Projeto e Implementação de Software**: Projeto e desenvolvimento (programação) do software atendendo às especificações.
3. **Validação de Software**: Verificação se o que foi construído atende as solicitações do cliente.
4. **Evolução de Software**: Evolução do software para que acompanhe as alterações solicitadas pelo cliente.

**Cada atividade** do **Processo Genérico de Software** é composta por um **Conjunto de Atividades** da **Engenharia de Software**.

Pressman (2016) afirma que uma **Metodologia Genérica da Engenharia de Software** é composta de **cinco atividades**, que são:

1. **Comunicação**: Com a intenção de entender os objetivos do projeto, a comunicação entre os envolvidos é a primeira ação primordial, para entender os requisitos (as funcionalidades) do software a ser realizado.
2. **Planejamento**: É realizado um “mapa”, um plano de projeto do software a ser realizado, descrevendo as tarefas técnicas, os riscos, os recursos, os produtos resultantes e um cronograma de trabalho (para acompanhar o desenvolvimento do software).
3. **Modelagem**: São criados modelos (diagramas) para melhor entendi­mento das necessidades do software, os modelos são utilizados para realizar a codificação do software e para validação das partes envol­vidas no projeto.
4. **Construção**: Realização da codificação (baseada nos modelos criados anteriormente), nesta fase também são realizados os testes para validar os códigos de programação gerados.
5. **Entrega**: O software é entregue parcialmente ou na sua totalidade, onde o cliente realiza testes e fornece um *feedback*; nesta fase são realizadas adaptações e correções no software por um
6. determinado período (acordado entre as partes).

Pressman (2016) destaca que as **Atividades Metodológicas** (citadas anteriormente) devem ter uma **série de tarefas** que darão suporte no acompa­nhamento e controle do projeto, controlando os riscos, fazendo revisões técnicas etc.

Entretanto, **a sequencialidade ou não de cada atividade genérica** pode ser dividida de acordo com o **Fluxo de Processo,** esse fluxo é usado para **descrever** como as atividades metodológicas de cada Processo são **organizadas**.

Os Fluxos de Processos podem ser:

1. **- Fluxo de Processo Linear**: Caracteriza-se por realizar em forma SEQUENCIAL as cinco atividades metodológicas apontadas por Pressman (2016):
2. {Comunicação -Planejamento -Modelagem -Construção -Entrega}
3. **- Fluxo de Processo Interativo**: Possui como característica a **repetição** de uma ou mais atividades antes de avançar para a próxima atividade.
4. **- Fluxo de Processo Evolucionário**: As atividades são executadas de **modo CIRCULAR,** cada ciclo envolve as **cinco atividades**, gerando uma versão mais completa (é um processo **incremental**).
5. **- Fluxo de Processo Paralelo**: As atividades são realizadas de **forma paralela**, onde **duas ou mais** **atividades** podem ser executadas **simul­taneamente,** por exemplo, a atividade de Comunicação pode ocorrer em paralelo com a atividade de Análise.

**A existência de um Processo de Software não garante a qualidade do software** e muito menos que o software será entregue no prazo combinado, também não é certo que as funcionalidades do software estarão de acordo com o que o cliente solicitou.

A **qualidade** do software produzido é direta­mente **influenciada** pelos **padrões de qualidade** impostos durante os **Processos de software** (durante a produção do software) sendo necessário estabelecer procedimentos e padrões para garantir a qualidade dos Processo**s.**

**O Processo de Software** deve ser **avaliado** para certificar que ele atenda a um conjunto de critérios básicos.

**Abordagens de Avaliação e Aperfeiçoamento dos Processos de Software:**

**(i) A abordagem SCAMPI:** (Método Padrão CMMI de Avaliação para Aperfeiçoamento de Processos da CMMI - *Standard CMMI Appraisal Method for Process Improvement*) fornece um modelo de **Avaliação do Processo** em cinco etapas:

{**Início-Diagnóstico-Estabelecimento-Atualização -Aprendizado}**

Define regras para assegurar a objetividade na classificação das avaliações, e ainda:

1. Ajuda a coletar e reunir evidências por meio de apresentações, documentos e entrevistas.
2. Transforma em anotações todas as evidências do que foi observado.
3. Converte as anotações em declarações de acertos ou falhas quando comparadas às práticas do CMMI.
4. Converte as declarações em descobertas preliminares.
5. Obtém a validação das descobertas preliminares e transforma as descobertas preliminares em definitivas.
6. CMMI (Modelo de Capacidade e Maturidade Integrado *- Capability Maturity Model Integration)* é um **conjunto de práticas** que orientam a implementação de uma série de atividades com o objetivo de alcançar uma meta pré-estabelecida, aumentando o **amadurecimento** organi­zacional e auxiliando a obter os resultados esperados pela área de TI.
7. Existem três modelos diferentes do CMMI:
8. CMMI *for Development*: apresenta melhores práticas para **desen­volver** produtos e serviços.
9. CMMI *for Acquisition:* mostra as melhores práticas para **adquirir** produtos e serviços.
10. CMMI *for Services:* indica as melhores práticas para **entregar** serviços.

(ii) **A abordagem CBA IPI**: (CMM *Based Appraisal for Internal Process Improvement* - Avaliação para o Aperfeiçoamento do Processo Interno baseado na CMM (*Capability Maturity Model* - Modelo de Maturidade em Capacitação), fornece uma técnica de diagnóstico para avaliar a maturidade relativa de uma organização de software.

O método possui a capacidade de identificar pontos fortes e fracos dos processos e viabiliza a possibilidade de priorização das melhorias mais relevantes.

(iii) **Abordagem SPICE (ISO/IEC15504:** É um padrão que define um conjunto de requisitos para avaliação do Processo de Software, possui a finalidade de auxiliar as organizações no desenvolvi­mento de uma avaliação objetiva da eficácia de um processo de qualquer software.

Este método fornece uma estrutura para a avaliação de Processos de Software e esta estrutura pode ser empregada nas organizações envolvidas na produção de um software.

(iv) **A abordagem ISO 9001:2000:** É um padrão genérico aplicável a qualquer organização que precise aplicar um padrão global de qualidade em seus produtos, sistemas ou serviços. Existem vários modelos de referências para ajudar a garantir a qualidade e que são aplicáveis a um software, alguns destes modelos são:

**ISO/IEC 9126 - ISO9000 -ISO9001 - ISO/IEC12207.**

**A ISO9001** descreve um modelo de garantia de qualidade em projetos, instalações, desenvolvimento e assistência técnica. Esta norma pode ser aplicada especificamente na área de desenvol­vimento, fornecimento e manutenção de software por meio dos roteiros descritos na norma **ISO 9000-3.**

A **ISO/IEC 9126** descreve as características de um software de qualidade.

Para exemplificar as atividades de um **Processo de Software**, veja algumas atividades:

1. **Especificação**: São realizados o estudo de Viabilidade, a Elicitação de Requisitos, a Especificação de Requisitos e a Validação dos Requisitos.
2. **Projeto**: São definidas as estruturas modulares do software, as *inter­faces* gráficas e as estruturas de dados (do banco de dados).
3. **Implementação**: Tudo o que foi decidido nas atividades de Especificação e Projeto é passado para uma linguagem de programação e o banco de dados é criado.
4. **Validação**: São realizados testes para validar tanto os códigos dos programas quanto a verificação dos requisitos (se o software atendeu aos requisitos impostos pelo cliente).
5. **Manutenção e Evolução**: São atividades contínuas a fim de melhora­mento do software e inclusão de novos recursos.