Seção 3

**Modelos de processos de software**

Teremos uma introdução ao conceito de **Modelo de Processo de Software**.

Veremos os Modelos de Processos **Prescritivos** e os Modelos de Processos **Especializados**, além de uma introdução ao **Desenvol­vimento Agil** e seus principais métodos:

Um **Processo de Software** é constituído por várias atividades, cuja finali­dade é ter como resultado um **Produto de Software**.

Sommerville (2011) destaca que dentre as atividades do **Processo de Software** estão:

O Estudo da Viabilidade:

A Análise dos Requisitos:

A Especificação:

A Arquitetura de Software:

A Implementação (codificação):

Os Testes:

A Documentação:

O Suporte

O Treinamento

A Manutenção.

Existem **Atividades Genéricas** e que aparecem na maioria dos **Processos de Software** e que são apontadas por Sommerville (2011):

**Análise e Especificação**: São realizadas as definições sobre o Software (a ser produzido) e determinados seus requisitos (funcionalidades) e suas restrições.

**Projeto**: É realizada a alocação de recursos (Hardware e Software) e são identificadas e definidas as abstrações do funcionamento do Software.

**Implementação e Teste Unitário**: É todo o processo de codificação do Software (seu desenvolvimento realizado por Analistas e Programadores) é a fabricação do Software.

**Integração e Verificação**: É todo o processo de avaliação, correção e validação, considerando a qualidade final do Software.

**Operação e Manutenção**: Nesta etapa são considerados todos os processos de alterações realizadas no Software (após ele estar em funcionamento).

Para o Gerenciamento das Atividades de Processo de Software são utili­zados os **Modelos de Processos de Software**. Um Modelo de Processo de Software tem como objetivo propiciar **estabilidade, controle** e **organização** das atividades:

Existem diversos **Modelos de Processos de Softwares** que possuem características diferentes.

**Modelos de Processos Prescritivos:**

Para cada Modelo de Processo Prescritivo também é indicado um Fluxo de Processo (ou Fluxo de Trabalho) ou seja, como esse conjunto de elementos do processo está interligado.

Alguns dos **Modelos de Processos Prescritivos** são os seguintes:

**Modelo Cascata**: (CLASSICO - TOP-DOWN) (Estender ao longo de meses)

Possui enfoque sistemático e sequencial dos Processos, cada fase é iniciada somente após a conclusão da fase anterior. **FASES: Análise e Especificação – Projeto - Implementação e teste de unidade** - **Integração e Verificação -** **Operação e Manutenção;**

**Modelo Incremental**: ( Receberá “pedaços” do Software (versões)

O Modelo Incremental é um modelo iterativo que visa, a partir de requi­sitos iniciais, criar pequenas versões do Software, que vão sendo entregues ao cliente, e posteriormente expandir o Software em novas versões até o sistema ideal ser totalmente construído, segundo Sommerville (2011). Nesse modelo, uma versão é um incremento. Cada incremento (ou versão) incorpora parte da funcionalidade requisitada pelo cliente.

**Modelos Evolucionários:** (Aparecem dois Modelos: **Prototipação e Espiral**)

O Modelo de Processo Evolucionário produz uma versão cada vez mais completa do Software. Pressman (2016) afirma que esse tipo de modelo é iterativo e evolui ao longo do tempo, o que se alinha perfeita­mente a um o projeto do Software, pois os requisitos do negócio e do produto não são estáveis, eles mudam (evoluem) e o Software pode ser desenvolvido pensando nesta evolução.

**Prototipação:** Tem seu início quando são definidos os requisitos do Software pelo cliente e o Analista de Sistemas. Após isso, é criado um protótipo do Software para que o cliente avalie e realize

testes de funcionalidades de modo a verificar se atende suas necessi­dades (requisitos), e servindo de orientação aos desenvolvedores.

**O Modelo Espiral:** É iterativo como a prototipação, mas utiliza os aspectos sistemáticos e controlados do Modelo Cascata, O objetivo do Modelo Espiral é fornecer um rápido desenvolvimento de versão, que a cada ciclo possa gerar versões mais completas.

**Modelos Concorrentes :**

São utilizados em projetos que **envolvem diferentes equipes** de desenvolvimento.

Os planos de projeto devem ser considerados **documentos vivos** e a evolução de cada Processo deve ser **avaliada com frequência** e revisada levando em consideração as alterações.

São representados graficamente por **uma série de tarefas e técnicas maiores** e **estados associados** a elas e que são utilizados como um paradigma para o desenvolvimento de aplicações Cliente/Servidor.

Os **Modelos de Processos Concorrentes** podem ser aplicados a **diversos tipos** de desenvolvimento de Softwares e, diferentemente do Modelo Cascata, ele **não segue uma sequência** de atividades, mas estabelece uma **rede de ativi­dades** que se movimentam de uma atividade para outra.

**Modelos de Processos Especializados**

**-Modelo Baseado em Componentes:**

São utilizados em projetos de **Software de Prateleira** (Software de Linha), compreende aplicações de componentes de Software previamente empacotados (e vendidos em partes ou completo).

**-Modelo de Métodos Formais:**

Compreende um conjunto de ativi­dades que levam à **especificação matemática formal** do Software, fornecendo mecanismos para a descoberta e a eliminação de muitos problemas como a ambiguidade, incompletude e inconsistência.

**-Desenvolvimento de Software Orientado a Aspecto :**

Fornece um processo e **abordagem metodológica** para **definir, especificar, projetar e construir** aspectos.

**-Modelo de Processo Unificado:**

Conhecida também como RUP – (*Rational Unified Process*) aproveita as características **dos Modelos de Processos tradicionais (Prescritivos)**, mas implementa alguns princí­pios da **metodologia Ágil** (abordada mais adiante nesta seção);

- **Modelos de Processos Pessoal e de Equipe:**

O cerne do desenvolvi­mento de Software está diretamente ligado **a toda equipe de desen­volvimento**.

No Modelo de **Processo de Software Pessoal** é enfati­zada a **medição pessoal do que foi produzido** (do artefato gerado e a qualidade resultante).

O Modelo de **Processo de Software de Equipe** objetiva a criação de **uma equipe autodirigida**, que se organiza por si mesma com a finalidade de produzir um Software com alto padrão de qualidade.

Os **Modelos de Processos Especializados** utilizam muitas das carac­terísticas de um ou mais Modelos de **Processos Prescritivos** e são utili­zados quando existe a necessidade de uma abordagem mais especializada de Engenharia de Software.

**Modelos de Desenvolvimento Ágil.**

**A Metodologia Ágil**, que traz um formato **mais flexível e dinâmico** nos Processos de Softwares.

O Princípio do Desenvolvimento Ágil **é focado nas entregas,** priorizando também **a comunicação** entre os envolvidos de forma ativa e contínua para realizar **entregas incrementais** (procurando a satisfação do cliente).

Sommerville (2011) destaca os seguintes princípios do Desenvolvimento Ágil:

**Envolvimento do Cliente : O** cliente deve ter um forte envolvimento no processo de desenvolvimento do Software fornecendo os requi­sitos e avaliando o que foi desenvolvido.

**Entrega Incremental**: O Software é desenvolvido com incrementos, o cliente indica os novos requisitos que devem ser acrescidos.

**Pessoas e Não Processos**: A equipe possui liberdade de desenvolvi­mento, explorando ao máximo a capacidade dos desenvolvedores (não estão presos a Processos Prescritivos).

**Aceitar as Mudanças**: Os requisitos podem ser alterados, o projeto deve ser elaborando pensando nesta possibilidade.

**Manter a Simplicidade:** A complexidade deve ser banida, a equipe deve trabalhar de forma simples e ativa.

Nesse universo, dois Métodos Ágeis se destacam:

**XP** (*Extreme Programming*) :

O feed back é constante,

Abordagem de desenvolvimento incremental,

Comunicação entre os envolvidos,

Atividade metodológicas que precisam seguidas no Extreme Programming:

O Planejamento – O projeto – A Codificação – O Teste;

Desenvolvimento Padronizado- Trabalho em par com cliente;

**Scrum**.

Quando o projeto inicializa, são definidas as ideias e funcionalidades iniciais do produto a ser desenvolvido, estas ideias são chamadas Histórias e o conjunto de todas as Histórias forma o *Product Backlog*.

No Método Scrum, geralmente as reuniões são conduzidas pelo *Scrum Master* (o líder da equipe) que conduz o processo e realiza avaliações das respostas de cada integrante da equipe, detectando de forma precoce eventuais problemas, como atrasos ou dificuldade de entendimento de algum requisito.

Conforme Pressman (2016), no término do *Sprint* os requisitos são concluídos e o funcionamento é avaliado, melhorando o processo para a *Sprint* sequencial.

Cada Sprint se encerra com um incremento ao produto (ou *Product Backlog*).

Desenvolvimento iterativos e incremental;

Utilizados em Projetos Gerenciais ;

Define um conjunto de regras e práticas de gestão : Trabalho em equipe e comunicação melhorada;

Atividades metodológicas que possui: Requisitos, Análise, Projeto, Evolução, Entrega;

Em cada atividade ocorrem as seguintes principais tarefas:

*Backlog*: Lista com prioridades dos requisitos (das funcionalidades) do projeto, na qual um item pode ser adicionado ou eliminado a qualquer momento (essas são as alterações) e o gerente do produto deve registrar e atualizar as prioridades.

*Sprint*: São unidades de trabalho para atingir um requisito (estabele­cido no *Backlog*) e precisa ser ajustado dentro do *Time Box* (Janela de Tempo) para definir os prazos de entrega.

1 Reuniões: Existe, ainda, uma reunião de planejamento na qual o *Product Owner* (dono do produto) prioriza os itens do *Product Backlog* e a equipe seleciona as atividades que ela será capaz de implementar durante o *Sprint* que se inicia.

A Equipe seleciona as atividades que vão conseguir implementar durante o SRINT que se inicia;

2 Reuniões Scrum: Reuniões *Scrum*: são reuniões breves, geralmente, de 15 minutos, chamadas *Daily Meeting* e realizadas diariamente (geralmente no início da manhã), nesta reunião são realizadas três perguntas chaves (para cada integrante da equipe): (i) O que você realizou desde a última reunião de equipe?; (ii) Quais foram os obstáculos encon­trados?; (iii) O que planeja realizar até a próxima reunião?

Quando o projeto se inicializa são desenvolvidas as ideias e as funcionalidades do produto, são as histórias e o conjuntos de todas histórias forma o Product back Log;

A reuniões são conduzidas pelo Scrum Master;

No término do Sprint os requisitos são concluídos e o conhecimento é avaliado;

Cada Sprint se encerra com um incremento do Product Backlog;

O Manifesto Ágil possui quatro valores fundamen­tais, exemplificando:

1. 1 Primeiro: Indivíduos e suas interações são mais importantes que os processos e ferramentas.
2. 2 Segundo: Software que funciona é mais importante que uma documentação vasta.
3. 3 Terceiro: A colaboração do cliente é mais importante que negociação de contratos.
4. 4 Quarto: Responder às mudanças solicitadas é mais importante que seguir um plano.