Aluno: João Victor Póvoa França

Período: 4°

Professor: Alex Coelho

Metodologias ágeis

Palmas-TO

Metodologia ágeis

São os métodos usados hoje em dia, SCRUMP, XP, TDD, em que são as chamadas metodologias ágeis. Mas afinal o que são as metodologias ágeis? A metodologia ágil é uma forma de gerir projetos, em que se busca a otimização dos processos. Em vez de seguir um plano rígido — como na gestão tradicional —, na metodologia ágil, podemos fazer ajustes e melhorias ao longo do desenvolvimento de projetos.

Além da flexibilidade e da adaptação às mudanças, as principais características dos métodos ágeis são:

- **Iteratividade (repetição);**
- **Colaboração;**
- **Autonomia e empoderamento das equipes;**
- **Entrega de valor.**

Manifesto ágil:

O Manifesto Ágil é um documento que sintetiza os valores e princípios das metodologias ágeis. Ele enfatiza:

- **Indivíduos e interações sobre processos e ferramentas.**
- **Software funcionando sobre documentação abrangente.**
- **Colaboração com o cliente sobre negociação de contratos.**
- **Responder a mudanças sobre seguir um plano.**

Esses princípios são fundamentais para entender a mentalidade por trás das metodologias ágeis e guiar as equipes no desenvolvimento de software de forma eficaz e adaptativa.

Exemplos:

Elas valorizam a colaboração, a comunicação eficaz e o trabalho em equipe. Algumas das metodologias ágeis mais populares são *Scrum, Extreme Programming (XP) e Test-Driven Development (TDD)*.

- ***Scrum*: Hoje é um dos modelos mais conhecidos do mercado, e mais utilizado, o medo definido por *sprints*! Nessas sprints ainda sim temos os chamadas *dailys*, que são reuniões diárias para definir o processo que será feito durante o dia, discutir melhorias e sincronização de trabalho.**

1. ***Scrum Master:* O *Scrum Master* é o responsável por fazer a equipe cumprir com os métodos, atividades e práticas do Scrum. Ele é quem tira os obstáculos do processo, e facilita as reuniões do time. Serve como o líder de equipe, com quem vai adotar o Scrum de forma eficaz e entender os processos.**

2. ***Product Owner:***

   Responsável por aumentar o valor do produto, representando os interesses do cliente, ou os chamados *stakeholders*. É ele quem define, e prioriza os itens dos requisitos do projeto, chamado *backlog do produto*.

3. **Time de desenvolvimento:**

   Como o próprio nome diz, quem faz o trabalho mais técnico, são os responsáveis por entregar o produto com suas funcionalidades. são auto organizados e multifuncionais, colaborando em alcançar os objetivos em cada sprint.

Extreme Programming (XP): Este modelo, prioriza práticas que pensam nas entregas de software de alta qualidade, práticas como: programação em pares, desenvolvimento orientado a Testes (TDD), integração contínua, refatoração e melhoria constante, entre outras.

- **Programação em pares: Na programação em pares, dois programadores trabalham juntos em uma mesma tarefa, sentados lado a lado ou remotamente. Um dos programadores escreve o código enquanto o outro revisa cada linha de código conforme é escrita. Eles trocam de papéis regularmente.**
- **Integração contínua: é uma prática em que as alterações de código são integradas ao repositório compartilhado frequentemente, geralmente várias vezes ao dia. Cada integração é verificada automaticamente por um sistema de build e testes automatizados.**
- **Refatoração e melhoria constante: Refatoração é o processo de reestruturar o código existente sem alterar seu comportamento externo. Essa prática visa melhorar a legibilidade, a manutenibilidade e o desempenho do código.**

- **Test-Driven Development (TDD): No desenvolvimento orientado a testes, os testes automatizados são escritos antes do código de produção. Os testes definem o comportamento esperado do sistema e, em seguida, o código é escrito para passar nesses testes.**
- **Benefícios:**
  - **Melhor design de código: O TDD incentiva o desenvolvimento de código modular e bem estruturado.**
  - **Maior confiança: Os testes garantem que as alterações no código não quebrem funcionalidades existentes.**
  - **Menor taxa de bugs: Ao escrever testes para cada funcionalidade, os desenvolvedores identificam e corrigem erros mais cedo no processo de desenvolvimento.**