ENGENHARIA DE SOFTWARE

Manifesto Ágil

O Manifesto Ágil foi construído pensando em se opor aos métodos tradicionais de desenvolvimennto de software, que eram burocráticos e inflexíveis. Os quatro pilares do Manifesto Ágil são: - **Indivíduos e interações mais que processos e ferramentas -** Software em funcionamento mais que documentação abrangente - **Colaboração com o cliente mais que negociação de contratos -** Responder a mudanças mais que seguir um plano

Hoje em dias as metodologias ágeis são amplamente adotadas, promovendo a valorização da adaptação, colaboração e entrega contínua de valor.

O desenvolvimento ágil foca no trabalho em equipes pequenas, auto-organizadas e multidisciplinares, priorizando a comunicação direta e a transparência em todas as etapas do processo de desenvolvimento. As principais metodologias ágeis são Scrum, eXtreme Programming (XP), Kanban, dentre vários outros.

eXtreme Programming (XP)

Conforme um autor (Beck, 2000), o eXtreme Programming é uma "Metodologia ágil para equipes pequenas a médias desenvolvendo software com requisitos vagos ou que mudam freqüentemente".

Em linhas gerais, no XP a codificação é a principal tarefa e baseia-se em "revisão permanente do código, testes frequentes, participação do usuário final, refatoração contínua, refinamento contínuo da arquitetura, integração contínua, planejamento, projeto e reprojeto a qualquer hora".

Valores

Os valores são a forma de funcionamento fundamental da metodologia, sendo eles: 1. Comunicação: Baseia-se na promoção da comunicação constante entre os membros da equipe, formal ou informalmente. 2. Coragem: Aumentar a confiança do programador com testes, integração contínua e, principalmente, a

programação em pares[1], para permitir ter coragem para mudar o código. 3. Feedbacks: Práticas que garantem o feedback sobre a qualidade do código (com a realização de testes unitários, de integração ou mesmo funcionais) e o estado de desenvolvimento, permitindo maior agilidade na correção de erros e reavaliação de conceitos que já haviam sido determinados. 4. Respeito: Saber ouvir, ter empatia e respeitar uns aos outros. 5. Simplicidade: Menos complexidade.

Princípios

Os princípios guiam o desenvolvimento usando a metodologia, sendo eles: 1. **Feedback rápido**: Obtenção contínua de retorno sobre decisões que foram tomadas. 2. **Presumir simplicidade**: Tratar cada problema como se pudesse ser resolvido de forma simples, mesmo que haja complexidade. 3. **Mudanças incrementais**: Realizar pequenas mudanças visando uma construção aos poucos do código. 4. **Abraçar mudanças**: Estar aberto e preparado para mudanças sempre que necessários. 5. **Trabalho de qualidade**: Priorizar a excelência técnica em todas as etapas do trabalho.

Práticas

São utiliazadas para o desenvolvimento do sistema, sendo elas: 1. **Equipe Inteira**: Toda a equipe trabalha em conjunto como um time de futebol. 2. **Jogos de Planejamento (Plan Poker)**: Definição de prioridade, tempo e custo do projeto. 3. **Posse Coletiva**: Todo mundo é "dono" do código. 4. **Programação em Pares (Pair Programming)**: O código tem um piloto e co-piloto. 5. **Ritmo Saudável**: Entre 36h a 40h por semana, sem exaurir a equipe. 6. **Testes de Aceitação**: Junto ao cliente com validações constantes. 7. **TDD (Desenvolvimento Orientado a Testes)**: Testar antes de codificar. 8. **Integração Contínua**: Realizar a validação constante dos componentes do sistema. 9. **Refatoramento**: Melhorar sem mudar o objetivo principal. 10. **Padrões de Código**: Conjunto de padrões de projeto que servem para resolução de problemas recorrentes no processo de desenvolvimento. 11. **Testes de Usuário**: Validação com o usuário final sobre as funcionalidades. 12. **Design Simples**: Manter o código limpo (clean code).

Atividades

O XP organiza o desenvolvimento em quatro atividades fundamentais:

1. **Codificação** É considerada a principal atividade no XP. Através dela, ideias são convertidas em código funcional.
2. **Testes** Criação e execução de testes automatizados antes mesmo da codificação (TDD). Garantem que o código funcione conforme esperado e servem como documentação viva do sistema.
3. **Escuta** Refere-se à habilidade dos desenvolvedores de ouvir atentamente os requisitos e necessidades dos clientes. Compreender o problema é essencial para desenvolver uma solução adequada.
4. **Design** Foca na criação de estruturas simples e eficientes que suportem a funcionalidade atual sem desnecessária complexidade. A refatoração contínua ajuda a manter o design do código limpo e adaptável.

---

1. **PROGRAMAÇÃO EM PARES**: Enquanto um programador faz o código, o outro revisa continuamente. É uma das principais práticas do XP. ↩