METODOLOGIAS ÁGEIS

AULA 1

Prof.ª Mariana Lucia Kalluf Dakkache Leal

CONVERSA INICIAL

Manifesto ágil e métodos ágeis

Os métodos ágeis são fundamentados nos valores e princípios do manifesto ágil, com o objetivo de entregar softwares com celeridade e eficiência, destacando a comunicação, a colaboração, o feedback e a adaptação a mudanças. Nesta abordagem, estudaremos os conceitos de agilidade, avaliando as vantagens e desvantagens desse método. Veremos como funciona uma cultura organizacional ágil, bem como os benefícios de sua implantação. Aprenderemos ainda como o manifesto ágil foi construído, seus valores e princípios. Falaremos ainda sobre as metodologias que fazem parte dos processos ágeis, bem como sobre os ganhos obtidos com sua utilização. Apresentaremos, por fim, como as ferramentas *pair programming* e refatoração podem ser utilizadas para trazer melhorias nos softwares de uma gestão ágil.

TEMA 1 – O QUE É AGILIDADE

De acordo com Jeff Sutherland (2015, p. 23), cocriador do Scrum, "ser ágil é ser capaz de mover-se rápida e facilmente; ser flexível, ágil na tomada de decisões, capaz de mudar de direção rápida e eficientemente; ser ágil é ser adaptável."

No ambiente de gerenciamento e desenvolvimento de projetos, agilidade é a capacidade de uma equipe ou organização de se adaptar rapidamente a mudanças, ser flexível e responder de modo produtivo a ambientes complexos e em constante mudança. A agilidade é uma abordagem colaborativa e orientada a resultados, priorizando a entrega de valor ao cliente, a aprendizagem contínua e a melhoria iterativa.

Eder et al. (2010) definem agilidade como “a habilidade de se adquirir velocidade e flexibilidade no gerenciamento de projetos por meio da adoção de práticas de gestão adequadas ao ambiente e tipo de projeto”

Crédito: curiosity/Shutterstock.

O conjunto de princípios e valores que fundamentam a agilidade foram definidos no manifesto ágil, criado por especialistas de software em 2001. Esses princípios e valores enfatizam a interação e a colaboração entre pessoas e equipes, a entrega gradual e funcional de software, a capacidade de adaptação a mudanças, a habilidade responder rapidamente a feedbacks e o compromisso constante com a melhoria técnica.

Práticas e abordagens que visam a flexibilidade e a adaptação, como utilização de iterações curtas, feedbacks frequentes, auto-organização das equipes, colaboração intensiva, prototipagem rápida, priorização do valor do cliente, entre outras, definem a agilidade.

A agilidade permite respostas rápidas às mudanças de requisitos, prioridades e condições de mercado, possibilitando que a equipe de desenvolvimento de software entregue valor ao cliente com mais celeridade.

"A agilidade é um movimento global de negócios que permite que as empresas entreguem produtos e serviços de maneira mais rápida, barata e eficiente. O ágil representa uma mudança de paradigma da administração de projetos, da gestão de produtos e do pensamento em geral" (Sutherland, 2015)

Não é somente no desenvolvimento de software que podemos aplicar os conceitos de agilidade. Temos uma grande frente de aplicação com essa metodologia, como gerenciamento de projetos, gestão de produtos, desenvolvimento de produtos, marketing, recursos humanos, em uma abordagem eficaz para lidar com a complexidade e a mudança rápida nos negócios e no mercado.

Crédito: JUVART/Shutterstock.

De acordo com Sommerville (2011, p. 40), o objetivo das metodologias ágeis é “reduzir a burocracia do processo, evitando qualquer trabalho de valor incerto de longo prazo e evitar qualquer documentação que provavelmente nunca será usada”.

1.1 Vantagens da agilidade em projeto

- **Entrega constante de valor**: permite que as equipes entreguem valor de forma contínua, aumentando a confiabilidade do projeto e satisfação do cliente.
- **Flexibilidade:** permite que as equipes se adaptem a mudanças com mais rapidez, facilitando assim os ajustes necessários aos requisitos do projeto.
- **Maior colaboração:** enfatiza a colaboração intensa entre as equipes de projeto e os stakeholders, garantindo que o produto atenda às necessidades do cliente.
- **Melhoria contínua:** incentiva a busca constante pela melhoria, o que pode ajudar as equipes a identificar oportunidades de melhorias ao longo do ciclo de vida do projeto.
- **Melhor gestão de riscos:** permite uma melhor gestão de riscos ao longo do projeto, não somente em seu início, de modo que as equipes poderão resolver problemas de forma mais rapidamente e com mais eficiência.

1.2 Desvantagens da agilidade em projeto

- **Menos documentação:** na maioria das vezes, enfatiza mais a entrega do produto do que a documentação do processo, podendo dificultar a rastreabilidade do projeto.
- **Necessidade de comunicação constante:** exige uma comunicação constante entre as equipes de projeto e os stakeholders, podendo ser mais desafiador em projetos com equipes distribuídas ou geograficamente dispersas.
- **Dificuldade em prever prazos e custos:** pode dificultar a definição de prazos e custos do projeto, visto que a entrega constante de valor pode acarretar mudanças frequentes nas prioridades e no escopo do projeto.

TEMA 2 – CULTURA ORGANIZACIONAL ÁGIL

A cultura organizacional diz respeito a normas e valores. De acordo com (Guerra, 2014, p. 54):

> A cultura organizacional ágil é baseada em princípios como transparência, inspeção e adaptação. Ela valoriza a colaboração, a comunicação aberta e a busca constante por melhorias. A cultura ágil encoraja a experimentação e o aprendizado contínuo, permitindo que a equipe e a organização se adaptem rapidamente às mudanças do mercado e do ambiente em que atuam. Essa cultura é essencial para que a organização possa se tornar mais ágil e competitiva, oferecendo soluções inovadoras e entregando valor aos seus clientes.

Em linhas gerais, a cultura de uma empresa corresponde aos seus princípios e valores compartilhados, juntamente com as condutas dos indivíduos que fazem parte dela.

Crédito: Viktoria Kurpas/Shutterstock.

Uma cultura organizacional consolidada contribui para garantir resultados melhores, maior eficiência e maior engajamento dos funcionários, fazendo com que eles se sintam mais comprometidos com a missão da organização. Uma cultura ágil, que nos últimos anos vem crescendo em popularidade, é uma metodologia que pode se adaptar, rápida e facilmente, às mudanças do projeto, sendo ideal para empresas modernas.

Muito mais do que um conjunto de regras, a cultura ágil é uma mentalidade fundamentada em três pilares, como mostra a Figura 1.

Figura 1 – Pilares da cultura organizacional ágil

Todas as ações da empresa devem estar centralizadas nas pessoas, sejam elas clientes membros da equipe ou funcionários em geral. O objetivo principal da cultura organizacional ágil é garantir resultados e entregas constantes, sempre permitindo flexibilidade para responder às mudanças com celeridade.

A cultura organizacional ágil é essencial para o sucesso da implementação de práticas ágeis em uma organização, pois afeta a forma como as equipes trabalham, se relacionam e tomam decisões.

2.1 Características da cultura organizacional ágil

A cultura organizacional ágil é caracterizada por uma série de valores, princípios e práticas que visam promover a colaboração, a entrega contínua de valor, a experimentação e a melhoria contínua. Algumas das principais características da cultura organizacional ágil incluem:

- **Foco no cliente:** as equipes ágeis estão comprometidas em entender as necessidades do cliente e em fornecer soluções que atendam às suas demandas.
- **Colaboração:** valoriza a colaboração entre os membros da equipe, bem como com outras equipes e partes interessadas.
- **Autonomia e responsabilidade:** as equipes ágeis são autônomas e responsáveis por tomar decisões e gerenciar o seu próprio trabalho.

- **Flexibilidade:** é adaptativa e flexível, permitindo que as equipes se adaptem a mudanças e sejam capazes de responder rapidamente a novas demandas e oportunidades.
- **Experimentação:** incentiva a experimentação e a aprendizagem contínua, permitindo que as equipes testem e validem hipóteses rapidamente.
- **Transparência:** promove a transparência nas informações, nas atividades e nas decisões, permitindo que as equipes trabalhem mais assertivamente.
- **Melhoria contínua:** as equipes ágeis estão sempre buscando melhorar a sua forma de trabalhar, por meio de avaliações e retroalimentações frequentes.
- **Foco em resultados:** é focada em resultados e entrega de valor, permitindo que as equipes se concentrem no que é mais importante para o cliente e para a organização.

TEMA 3 – MANIFESTO ÁGIL: CONCEITOS, VALORES E PRINCÍPIOS

3.1 Conceitos

O Manifesto Ágil foi criado em fevereiro 2001 por um grupo de 17 desenvolvedores, que se reuniram em Snowbird, Utah, nos Estados Unidos, para discutir maneiras de melhorar a forma como o software era desenvolvido.

Crédito: Viktoria Kurpas/Shutterstock.

Nessa reunião, foram analisados os pontos em comum dos projetos que levam ao sucesso de suas metodologias. A partir dessas constatações, foi criado o Manifesto para Desenvolvimento Ágil de Software, mais conhecido como Manifesto Ágil.

No Manifesto Ágil (2001) os desenvolvedores destacam: “Estamos descobrindo maneiras melhores de desenvolver softwares fazendo-o nós mesmos e ajudando outros a fazê-lo”. O manifesto foi assinado pelos 17 desenvolvedores, que se intitulavam Aliança Ágil:

- Alistair Cockburn: criador da Metodologia Ágil Crystal
- Andrew Hunt: coautor de “O Programador Pragmático”
- Arie van Bennekum: Integrated Agile
- Brian Marick: cientista da computação e autor de livros sobre programação
- Dave Thomas: programador e coautor de “*The Pragmatic Programmer*”.
- James Grenning: autor de “*Test Driven Development*”
- Jeff Sutherland: inventor do Scrum.
- Jim Highsmith: criador do Adaptive Software Development (ASD)
- Jon Kern: cofundador da Agile Alliance
- Ken Schwaber: cocriador do Scrum
- Kent Back: cocriador da eXtreme Programming (XP)
- Martin Fowler : desenvolvedor parceiro da Thoughtworks
- Mike Beedle: coautor de Desenvolvimento Ágil de Software com Scrum
- Robert C. Martin: o “Uncle Bob”
- Ron Jeffries: cocriador da eXtreme Programming (XP)
- Steve Mellor: cientista da computação e um dos idealizadores da Análise de Sistema Orientado a Objetos (OOSA)
- Ward Cunningham: criador do conceito wiki

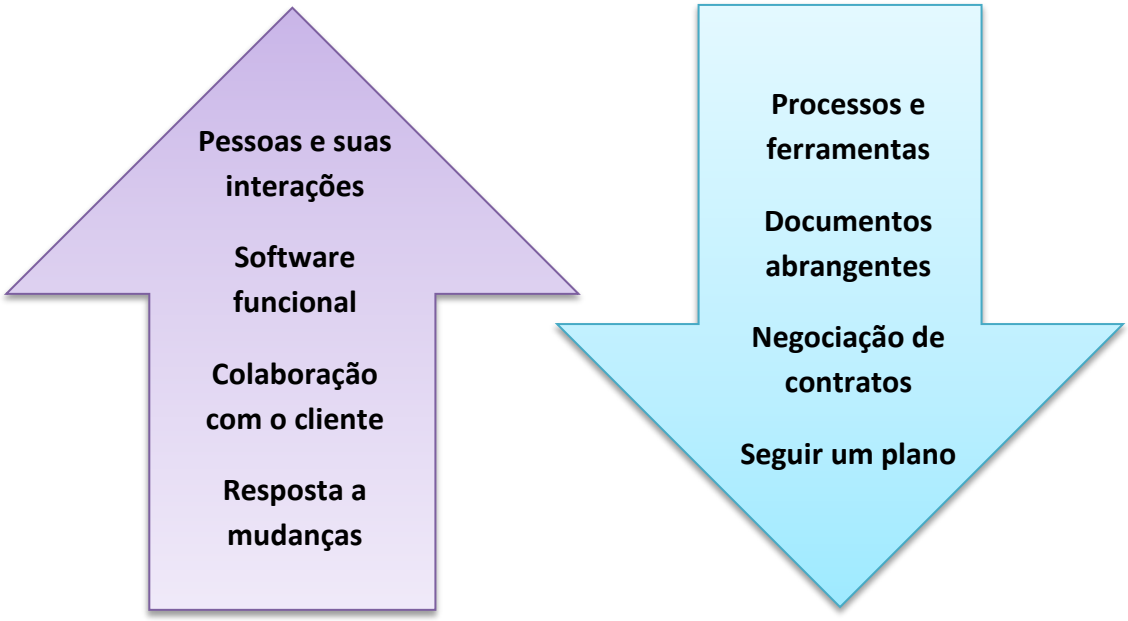
O Manifesto Ágil é uma declaração de valores e princípios essenciais para o desenvolvimento de software.

3.2 Valores

Os autores do Manifesto Ágil tiveram como objetivo demostrar aos profissionais envolvidos no desenvolvimento de um projeto que os fatores que

estão no lado esquerdo devem ser priorizados em relação aos fatores da direita. A figura a seguir traz a representação desses valores.

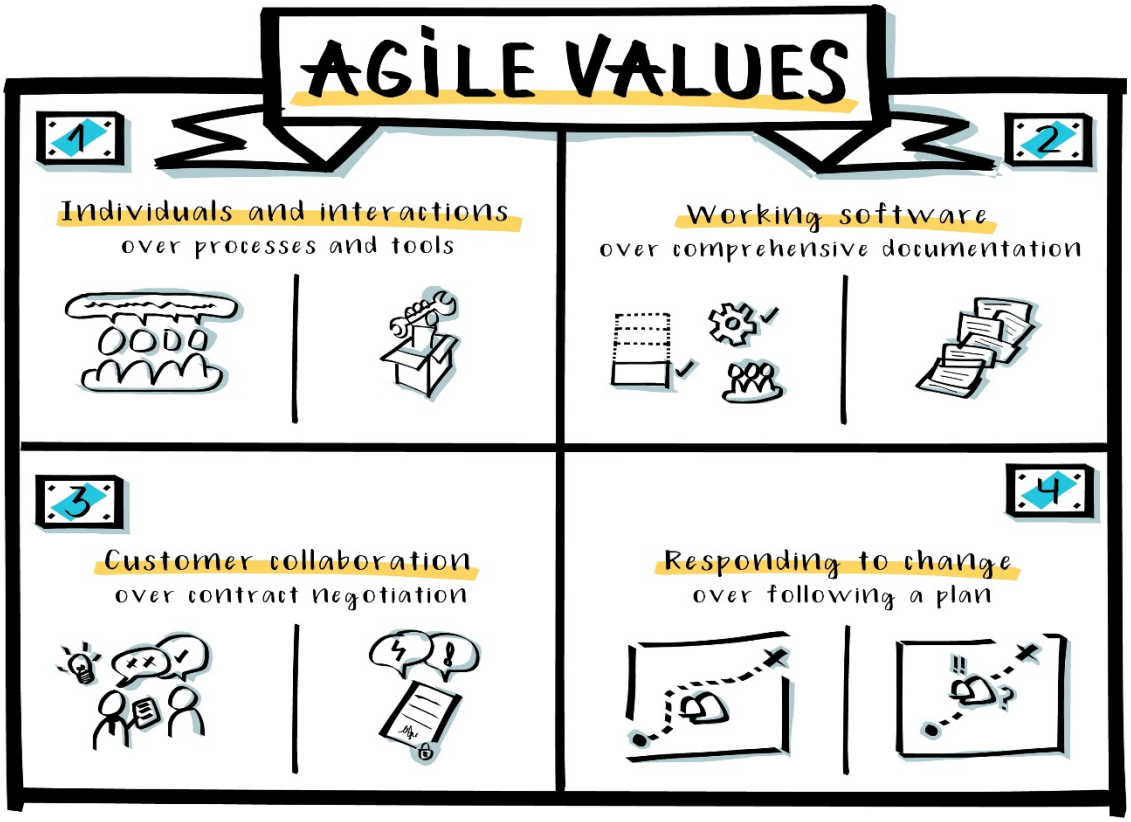
Figura 2 – Valores ágeis



Os quatro valores definidos no manifesto ágil, apresentados a seguir, precisam ser compreendidos completamente, para um desenvolvimento de software adequado e para a entrega de produtos com qualidade.

- **Indivíduos e interações mais que processos e ferramentas:** o desenvolvimento de software é uma atividade que envolve pessoas. Quando temos uma interação com qualidade entre as pessoas, evitamos ou resolvemos problemas crônicos de comunicação. É essencial que processos e ferramentas sejam descomplicados e funcionais.
- **Software em funcionamento mais que documentação abrangente:** o melhor indicativo de que uma equipe efetivamente construiu algo é obtido com o software em funcionamento, que também é o que os clientes procuram. A documentação tem a sua importância no projeto, mas deve ser elaborada somente com o essencial, devendo necessariamente agregar valor na construção do software.
- **Colaboração com o cliente mais que negociação de contratos:** é imperativo atuar em parceria com o cliente, e não em um cenário de um contra o outro. É preciso buscar a colaboração, decidir em conjunto, trabalhando em equipe, unindo forças em prol de um objetivo comum.

- **Responder a mudanças mais que seguir um plano:** o ambiente de desenvolvimento de software e produtos gera muitas incertezas e mudanças. Não devemos nos comprometer com planos extensos e cheio de hipóteses. Devemos aprender com conhecimentos adquiridos e feedbacks recebidos, ajustando os planos constantemente.



Crédito: AmaiaL/Shutterstock.

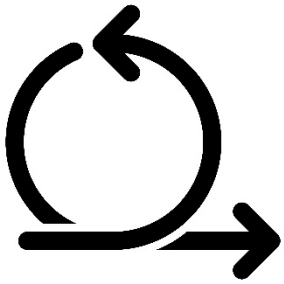
3.3 Princípios

Além dos quatro valores propostos pelo (Manifesto Ágil, 2001), foram propostos 12 princípios norteiam o desenvolvimento ágil:

- **Princípio Ágil 1: Nossa maior prioridade é satisfazer o cliente, por meio da entrega antecipada e contínua de software de valor**. A equipe do projeto elabora um Produto Mínimo Viável (MVP), para que o cliente teste e forneça um feedback. Com esse resultado da validação, é possível retomar o desenvolvimento do produto, melhorando as versões futuras.
- **Princípio Ágil 2: Aceitar mudanças de requisitos, mesmo no fim do desenvolvimento. Processos ágeis se adequam a mudanças, para que o cliente possa tirar vantagens competitivas**. O monitoramento do mercado, das percepções do cliente e de demais fatores relevantes pode

alterar a direção do produto. Cabe à equipe do projeto receber esses acionamentos e ajustar os planos para atender às necessidades do cliente e do negócio.

- **Princípio Ágil 3: Forneça software de trabalho com frequência, de algumas semanas a alguns meses, com preferência pela menor escala de tempo.** O ciclo de desenvolvimento ágil quebra o produto em entregas menores com prazos definidos. Essa entrega frequente propicia à equipe de desenvolvimento a validação constante de ideias, com a possibilidade de melhorias contínuas.



Crédito: GzP_Design/Shutterstock.

- **Princípio Ágil 4: Pessoas relacionadas a negócios e desenvolvedores devem trabalhar em conjunto e diariamente, durante todo o curso do projeto**. Na metodologia ágil, as equipes são multifuncionais e incluem as pessoas do produto. Com isso, é possível ter uma visão técnica e uma visão de negócio.
- **Princípio Ágil 5: Construir projetos ao redor de indivíduos motivados, dando a eles o ambiente e o suporte necessários, e confiando que farão o seu trabalho**. A capacitação de pessoas e equipes é uma parte essencial da metodologia ágil. A construção de uma equipe de projeto é uma atividade que requer atenção, para que sejam incluídas pessoas com capacidades adequadas. As responsabilidades devem ser amplamente definidas antes do início do projeto.
- **Princípio Ágil 6: O Método mais eficiente e eficaz de transmitir informações para (e por dentro de) um time de desenvolvimento é por meio de uma conversa cara a cara**. Incentivar a interação humana, a conversa cara a cara, mesmo que seja realizada por meio de uma videoconferência. O objetivo desse princípio é encorajar a comunicação

efetiva e em tempo real entre a equipe do projeto. Essa comunicação pode ocorrer em reuniões, demonstrações ou sessões colaborativas do projeto.

Crédito: elenabsl/Shutterstock.

Crédito: elenabsl/Shutterstock.

- **Princípio Ágil 7: Software funcional é a medida primária de progresso**. O tempo de projeto deve ser gasto com o desenvolvimento do software, e não com a escrita detalhada e perfeita de uma documentação. A equipe ágil deve ter em mente que é necessário projetar e disponibilizar ao cliente um MVP (Produto Mínimo Viável) para receber as devolutivas do cliente e com isso ajudar a evoluir o produto.

- **Princípio Ágil 8: Processos ágeis promovem um ambiente sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter passos constantes indefinidamente**. Identificar sempre o volume de trabalho que cada integrante da equipe é capaz de suportar. Depois, é preciso definir as atividades do desenvolvedor. Não devem mais ser adicionadas novas atividades até que um ciclo de desenvolvimento seja iniciado. Os gerentes de produto devem monitorar as atividades adicionadas, para que sejam entregues conforme proposto, evitando mudanças do produto.
- **Princípio Ágil 9 Contínua atenção à excelência técnica e ao bom design aumenta a agilidade**. A equipe precisa estar atenta para manter a organização do código, evitando assim a necessidade de ajustes futuros.
- **Princípio Ágil 10: Simplicidade é a arte de maximizar a quantidade de trabalho que não precisou ser feito**. Os princípios ágeis propõem que primeiramente devemos fazer o que tem mais impacto. Devemos desenvolver somente o que tem propósito e é estratégico para o cliente.
- **Princípio Ágil 11: As melhores arquiteturas, requisitos e designs emergem de times auto-organizáveis**. A ideia é formar equipes auto-organizadas, sem uma cadeia hierárquica. Equipes que tomam decisões em grupo, que detêm o controle e as responsabilidades por seu projeto. Esse princípio está ligado ao valor “Indivíduos e interações mais que processos e ferramentas”, e tem por objetivo habilitar as equipes no trabalho conjunto.
- **Princípio Ágil 12: Em intervalos regulares, o time reflete para ficar mais efetivo. Assim, ocorrem ajustes com otimização do comportamento de acordo**. Buscar sempre a melhoria contínua dos processos e equipes.

> Esses princípios, se bem compreendidos pelo cliente e pelo time, promovem uma mudança de atitude. O cliente consegue enxergar valor mais rapidamente nas entregas frequentes do software e, à medida que visualiza a solução, consegue refletir sobre alternativas e prioridades. O time trabalha mais motivado, porque consegue enxergar valor no seu trabalho que, por meio de feedback constante, aprimora continuamente. O bom relacionamento melhora para ambos, visto que a confiança se faz cada vez mais presente. (Wildt et al., 2014, p. 13)

TEMA 4 – PROCESSOS ÁGEIS

Processos ágeis são técnicas de gerenciamento de projetos e desenvolvimento de software que priorizam a adaptação, a colaboração e a entrega iterativa e incremental. Foram desenvolvidos como uma opção menos burocrática em comparação aos métodos tradicionais de gestão de projetos.

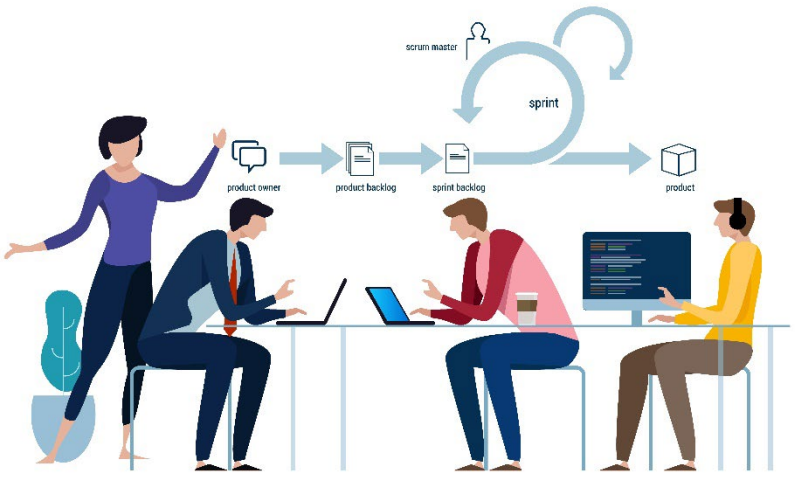
Esses processos têm por objetivo permitir que as equipes trabalhem com maior adaptabilidade e flexibilidade, não se atendo a um plano rígido e pré-definido. As equipes trabalham com pequenos ciclos de desenvolvimento e entregam versões do produto que evoluem a cada ciclo.

Segundo Sommerville (2011, p. 40), os métodos ágeis partilham uma série de princípios, com base no Manifesto Ágil, razão pela qual têm muito em comum. Embora sejam fundamentados na ideia de desenvolvimento e entrega incremental, propõem diferentes processos para alcançar tal objetivo.

> Os processos ágeis têm como principal objetivo criar um ambiente que permita a entrega rápida e eficiente de software, priorizando a comunicação, a colaboração, o feedback e a adaptação a mudanças. Dentre os processos ágeis mais utilizados, podemos citar o Scrum, o Extreme Programming (XP) e o Kanban, cada um com suas particularidades e foco em diferentes aspectos do desenvolvimento de software. (Cohn, 2010, p. 20)

Os processos ágeis apresentam diversas metodologias. As mais populares são:

- **Scrum:** metodologia ágil mais popular, baseada em iterações de trabalho de curto prazo, chamadas de Sprint, geralmente com duração de 1 a 4 semanas. Em cada um desses ciclos, as equipes de desenvolvimento de software planejam, desenvolvem, testam e entregam incrementos de software funcional. O Scrum define os papéis da seguinte forma: Scrum Master, Product Owner e Time de Desenvolvimento. Utiliza artefatos, como backlog do produto e backlog da sprint, para gerenciar o trabalho.



Crédito: Bakhtiar Zein/Shutterstock.

- **Kanban:** metodologia de gestão que se concentra na visualização do fluxo de trabalho por meio do quadro Kanban, dividido por colunas que representam as etapas do processo, e com os cartões que representam as tarefas em andamento.

Crédito: Bakhtiar Zein/Shutterstock.

- **Lean:** baseia-se na filosofia Lean de gestão de produção, buscando minimizar desperdícios e maximizar o valor para o cliente. O Lean Agile busca identificar e eliminar retrabalho, espera e excesso de processos, entendidos como desperdícios em processos de desenvolvimento de software.

Crédito: Sabelskaya/Shutterstock.

- **Extreme Programming (XP):** foi criada em 1990, em busca da qualidade do código, com colaboração intensiva e satisfação do cliente. São práticas comuns dessa metodologia: programação em pares (Pair programming), testes automatizados, integração contínua, design simples e reuniões diárias de equipe.

Crédito: mentalmind/Shutterstock.

- **Crystal:** foi criada por Alistair Cockburn, sendo indicada para equipes com até 8 pessoas, para projetos de pequeno e médio porte. Valoriza a entrega incremental, a simplicidade e a qualidade do produto.

Crédito: aurielaki/Shutterstock.

Essas são apenas algumas das muitas metodologias ágeis disponíveis. Cada uma delas apresenta as suas próprias características e práticas específicas, mas todas compartilham os princípios e valores do Manifesto Ágil, como colaboração, flexibilidade, entrega incremental e busca pela qualidade do software. As metodologias ágeis são amplamente utilizadas em projetos de desenvolvimento de software para promover a agilidade, a adaptabilidade e o sucesso dos projetos.

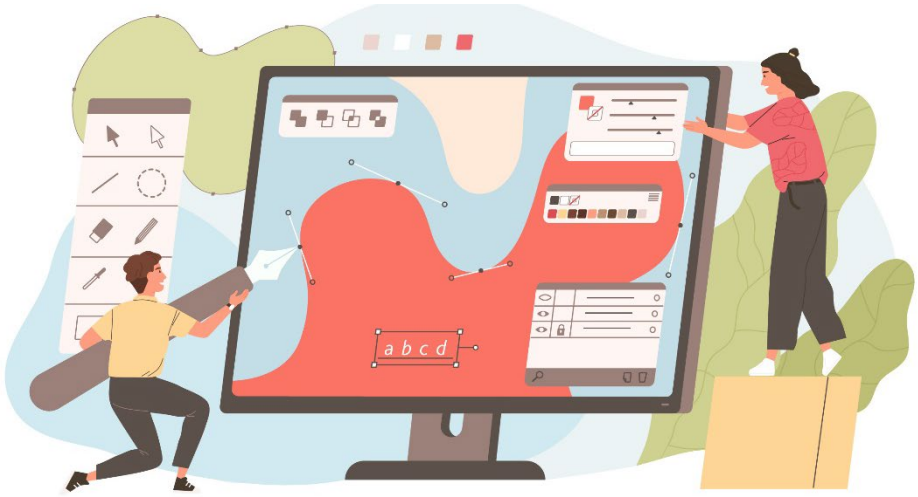
TEMA 5 – FERRAMENTAS *PAIR PROGRAMING* E REFATORAÇÃO

5.1 *Pair programing*

De acordo com Fowler (1999, p. 199), o *pair programming* “é uma prática na qual dois programadores trabalham juntos em um computador para desenvolver softwares. Envolve discutir o que fazer, projetar, codificar, testar e, de certa forma, constantemente revisar o trabalho um do outro."

O *pair programming*, ou programação em pares, é fundamentado em colaboração e trocas contínuas de ideias e conhecimentos da dupla. O desenvolvedor que escreve o código é chamado de piloto, enquanto o desenvolvedor que está acompanhando, questionando, contribuindo com

alternativas e detectando possíveis problemas é chamado de observador ou navegador.



Crédito: GoodStudio/Shutterstock.

Segundo Guerra (2015), o *pair programming* “tem demonstrado uma série de benefícios para o desenvolvimento de software, como a melhoria da qualidade do código, a redução de erros, a melhoria na produtividade da equipe, o compartilhamento de conhecimento e a melhoria na comunicação e colaboração entre os membros da equipe."

Com o uso dessa técnica, os desenvolvedores conseguem identificar problemas com menos tempo de análise de código, buscando soluções mais eficazes. O compartilhamento de ideias de forma constante e colaborativa reduz a criação de códigos redundantes ou desnecessários. Com isso, as soluções são mais elegantes e eficientes.

A curva de aprendizado para novos membros pode ser reduzida com a utilização da programação em pares. Quando os desenvolvedores trabalham em parceria, conseguem se familiarizar mais rapidamente com o código, a arquitetura e a lógica de programação.

O trabalho em pares permite que os desenvolvedores aprendam novas técnicas e desenvolvam novas habilidades. Com isso, ficam mais motivados e até mais desafiados em comparação a situações em que trabalham sozinhos. Entretanto, temos casos em que os desenvolvedores não se adaptam à rotina de trabalhar juntamente com outro programador, pois podem se sentir intimidados e desconfortáveis.

De acordo com Lopes e Marques (2014), as vantagens do *pair programming* são:

- Aumento da qualidade do código, já que dois desenvolvedores estão trabalhando juntos para encontrar e corrigir erros;
- Maior produtividade, pois dois desenvolvedores podem trabalhar em diferentes tarefas simultaneamente, o que aumenta a quantidade de trabalho concluído;
- Melhora na comunicação e colaboração entre os membros da equipe, reduzindo mal-entendidos e conflitos;
- Maior aprendizado e compartilhamento de conhecimento entre os desenvolvedores.

Por outro lado, as desvantagens apontadas por Lopes e Marques (2014) incluem:

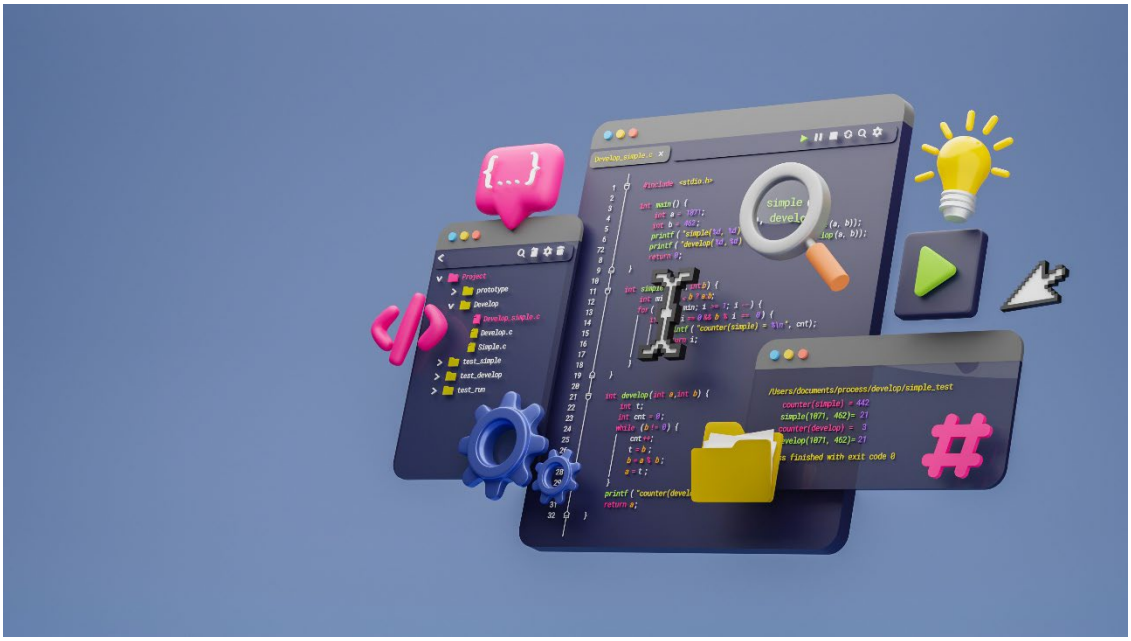
- Maior custo, já que duas pessoas estão trabalhando na mesma tarefa;
- Possibilidade de conflitos e divergências entre os desenvolvedores, especialmente em caso de comunicação e colaboração deficientes;
- Possibilidade de desigualdade na contribuição dos desenvolvedores, se um deles dominar mais a tarefa ou o assunto em questão.

5.2 Refatoração

De acordo com (Fowler, 2019, p. 3)

> A refatoração é uma técnica de melhoria contínua de código que visa torná-lo mais legível, compreensível e fácil de manter. Ela consiste em fazer pequenas alterações no código sem mudar seu comportamento externo, com o objetivo de eliminar código duplicado, simplificar estruturas complexas, melhorar a modularidade e a flexibilidade do sistema. A refatoração é uma prática fundamental para manter a qualidade do código ao longo do tempo e evitar o acúmulo de débito técnico.

Na agilidade, a refatoração é uma prática recorrente, que está incorporada ao processo de desenvolvimento de software. Deve ser realizada gradualmente, à medida que o software for desenvolvido. A intenção dessa técnica é que os desenvolvedores realizem as correções no código com frequência, no ritmo de sua evolução, mantendo assim a sua qualidade.



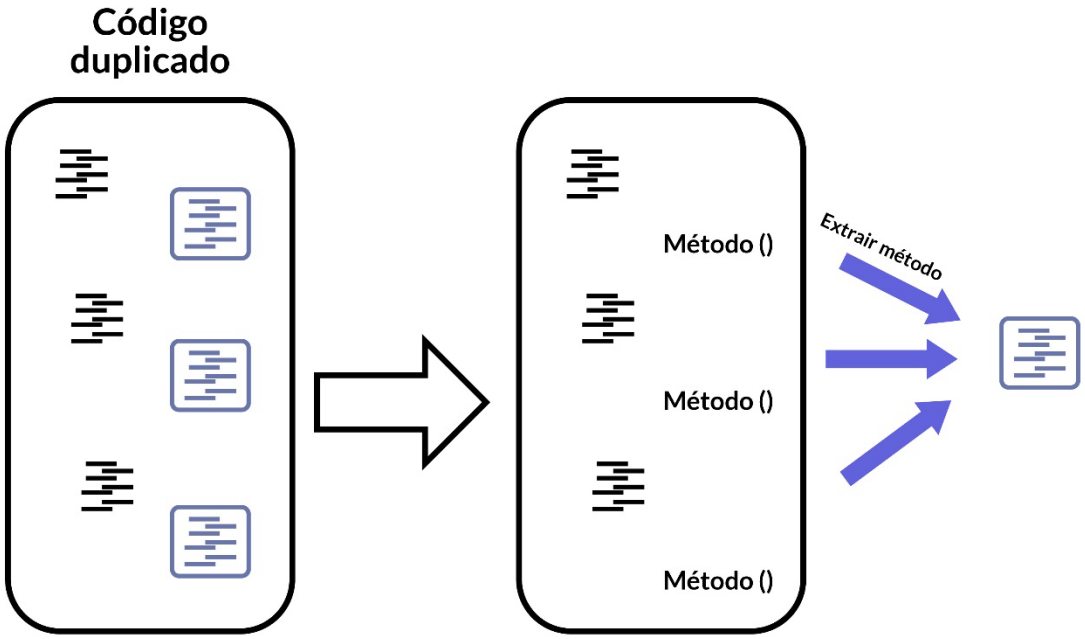
Crédito: Chaosamran_Studio/Shutterstock.

Segundo Martin (2009, p. 205), “O objetivo da refatoração é melhorar o design do software existente, tornando-o mais fácil de entender, mais eficiente, com menor acoplamento e maior coesão, sem alterar seu comportamento externo." Fundamentada em técnicas e princípios de engenharia de software, tem por objetivo aprimorar a qualidade do código fonte, evitando o acúmulo de falhas no software.

É de suma importância que a refatoração seja realizada de forma estruturada e com cautela. É essencial programar a realização de testes para assegurar que alterações no código não acarretem novos problemas.

"Como um processo de melhoria contínua, a refatoração pode incluir a reorganização do código, a eliminação de duplicação, a melhoria da legibilidade, a simplificação de algoritmos complexos, a melhoria da estrutura de dados, entre outros ajustes" (Fowler, 2018, p. 17).

Figura 3 – Refatoração de código



"Refatoração pode ser uma atividade arriscada. Pode ser tentador fazer grandes mudanças que resultam em melhorias significativas, mas também podem introduzir novos problemas. Refatorações menores e mais frequentes são menos arriscadas" (Beck, 2002, p. 110).

FINALIZANDO

Nesta abordagem, tratamos de conceitos fundamentais sobre agilidade, incluindo a cultura organizacional ágil e os valores e princípios do Manifesto Ágil. Além disso, apresentamos alguns dos processos ágeis mais utilizados, como Scrum, XP, FDD, TDD e Crystal, avaliando a importância de ferramentas como o *pair programming* e a refatoração no desenvolvimento de software ágil. O conhecimento desses conceitos e práticas é fundamental para um desenvolvimento de software ágil, eficiente e eficaz, permitindo que equipes de desenvolvimento entreguem valor incrementalmente, respondendo com rapidez em cenários de mudanças.

REFERÊNCIAS

BECK, K. **Test-Driven Development**: By Example. Addison-Wesley Professional, 2002.

COHN, M. **User Stories Applied**: For Agile Software Development. 2. ed. São Paulo: Pearson Education, 2010.

EDER, S. et al. Estudo exploratório do conceito agilidade: modelo teórico para aplicação no gerenciamento ágil de projetos. In: SIMPÓSIO BRASILEIRO DE ENGENHARIA DE PRODUÇÃO, 17., 2010, Bauru. **Anais**... Bauru: UNESP, 2010.

FOWLER, M. **Refactoring**: aperfeiçoando o design de códigos existentes. São Paulo: Novatec Editora, 2018.

_____. **Refactoring**: Improving the Design of Existing Code. Addison-Wesley Professional, 1999.

_____. **Refatoração**: aperfeiçoando o design de códigos existentes. São Paulo: Pearson, 2019.

GUERRA, E. **Cultura ágil**: manifesto ágil, lean, scrum e XP. São Paulo: Casa do Código, 2014.

_____. Pair programming: uma revisão sistemática. **Revista de Informática Teórica e Aplicada – RITA**, v. 22, n. 2, p. 45-56, 2015.

LOPES, H.; MARQUES, R. **Pair programming**: vantagens e desvantagens. Lisboa, 2014. Disponível em: <http://paginas.fe.up.pt/~arestivo/page_uploads/PairProgramming.pdf>. Acesso em: 24 maio 2023.

MANIFESTO ÁGIL. 2001. Disponível em: <http://www.manifestoagil.com.br/>. Acesso em: 24 maio 2023.

MARTIN, R. C. C. C. **A Handbook of Agile Software Craftsmanship**. Pearson Education, 2009.

SOMMERVILLE, I. **Engenharia de software**. 9. ed. São Paulo: Pearson Prentice Hall, 2011.

SUTHERLAND, J. **Scrum**: a arte de fazer o dobro do trabalho na metade do tempo. São Paulo: LeYa, 2015.

WILDT, D. **eXtreme Programming**. São Paulo: Casa do Código, 2014.