Metodologia Ágil FDD (Feature-Driven Development)

# Introdução

Feature-Driven Development (FDD) é uma metodologia Ágil de Desenvolvimento de Software que enfatiza a entrega constante de funcionalidades. Baseado no método Coad, idealizado por Peter Coad, o FDD foi desenvolvido entre os anos de 1997 e 1999 pelo estrategista de tecnologia Jeff De Luca. A principal abordagem do FDD é a entrega constante de funcionalidades (features) que juntos formam um todo. Esta abordagem contrasta com métodos tradicionais de desenvolvimento, que muitas vezes envolvem ciclos longos de planejamento e desenvolvimento antes que qualquer funcionalidade utilizável seja entregue.

# Visão Geral do FDD

A metodologia é composta por um método iterativo e incremental visando a entrega constante para o cliente. A estrutura do FDD se baseia em cinco atividades básicas que são seguidas para desenvolver um projeto de software:

## **Desenvolvimento de um modelo global**

Nesta fase, a equipe se concentra em projetar um modelo global e uma visão de alto nível do sistema, é nesta etapa que os principais conceitos do sistema são identificados e organizados. A modelagem do domínio envolve analistas de negócios, desenvolvedores e stakeholders do projeto. O objetivo é estabelecer um entendimento comum da natureza do problema, bem como das principais entidades e conexões que compõem o sistema. Os diagramas e descrições gerados pela modelagem do domínio servem como base para o desenvolvimento de funcionalidades específicas.

## **Construir uma lista de funcionalidades**

A equipe cria uma lista detalhada de funcionalidades que precisam ser implementadas usando a visão geral do modelo. Cada funcionalidade é um pequeno componente do sistema que dá ao usuário um valor específico. As "ações" e "resultados", que são importantes para os usuários finais, são as características mais comuns. Uma funcionalidade pode, por exemplo, "calcular o total do pedido" ou "gerar relatórios de vendas mensais". A priorização da lista de funcionalidades é determinada pelo valor do negócio e pelas dependências entre as funcionalidades.

## **Planejar por funcionalidade**

Nesta etapa, a equipe planeja o desenvolvimento de cada função neste ponto. A definição de prazos e a distribuição de tarefas são parte disso. O planejamento por funcionalidade significa estabelecer um cronograma detalhado para quando cada funcionalidade será desenvolvida e disponibilizada. Ao criar o plano, a equipe deve pensar em coisas como complexidade técnica, disponibilidade de recursos e prioridades de negócios. O objetivo é garantir que o desenvolvimento seja feito de forma organizada e que as funcionalidades sejam entregues de acordo com as expectativas do cliente.

## **Designe por funcionalidade**

Antes de ser usado, cada funcionalidade é projetada minuciosamente. A identificação de classes, métodos e diagramas de sequência são parte disso. Para garantir que a implementação seja eficaz e que a funcionalidade atenda aos requisitos especificados, o design detalhado é uma etapa crucial. Quando o design está em andamento, a equipe identifica quaisquer outros componentes ou necessidades que possam afetar a implementação. Os diagramas de sequência mostram as interações e o fluxo de dados entre vários componentes do sistema, o que facilita a identificação de problemas ou melhorias no design.

## **Construir por funcionalidade**

Por fim, todas as funcionalidades são desenvolvidas e testadas. A implementação é feita em pequenas etapas, com o objetivo de fornecer funcionalidades completas e testáveis. O desenvolvimento iterativo permite que os stakeholders forneçam feedback contínuo e façam ajustes quando necessário. A escrita de código, testes unitários, integração com outros componentes do sistema e revisão de código são etapas necessárias para construir cada funcionalidade. O objetivo é garantir que todas as funcionalidades estejam funcionando corretamente e que quaisquer problemas sejam encontrados e resolvidos rapidamente.

# Benefícios do FDD

A metodologia FDD oferece uma variedade de vantagens para o desenvolvimento de software, como:

• **Foco na Entrega de Valor:** O FDD garante que cada iteração de desenvolvimento produza valor para o cliente ao se concentrar em funcionalidades específicas. Isso melhora a satisfação do cliente e ajuda a manter a relação entre o crescimento e as necessidades do negócio. Ao se concentrar nas funcionalidades, a equipe pode priorizar e gerenciar as mudanças, permitindo que eles respondam rapidamente a novas necessidades ou mudanças nos requisitos.

• **Processo Estruturado:** Em comparação com outras metodologias ágeis, o FDD oferece um processo claro e estruturado, o que pode ser útil para equipes que preferem uma abordagem mais organizada. A estrutura FDD ajuda a garantir que todas as etapas do desenvolvimento sejam cobertas e que a equipe siga práticas consistentes e repetitivas. Isso pode aumentar a qualidade do software e, simultaneamente, diminuir a probabilidade de erros ou retrabalho.

• **Melhoria Contínua:** A equipe pode melhorar continuamente o produto e adaptá-lo às necessidades do cliente por meio de iterações e entregas incrementais. A entrega contínua de funcionalidades permite que o cliente acompanhe o progresso do projeto e forneça feedback, o que é essencial para garantir que o produto final atenda às expectativas. A melhoria contínua inspira criatividade e melhores práticas em equipe.

# Desafios do FDD

Apesar dos benefícios, o FDD traz alguns problemas que precisam ser gerenciados corretamente:

* **Complexidade Inicial:** A fase inicial de modelagem e planejamento pode ser difícil e lenta e requer uma compreensão completa do domínio do problema. A colaboração estreita entre a equipe de desenvolvimento e os stakeholders é necessária para criar uma visão geral do modelo e uma lista de funcionalidades. Isso pode ser um obstáculo em projetos grandes ou complexos. Além disso, algumas equipes podem não gostar da necessidade de detalhamento nas primeiras fases.
* **Dependência de Especialistas:** Algumas equipes podem encontrar que a presença de especialistas em domínio e modelagem é necessária para que o FDD funcione bem. A modelagem do domínio e a criação de designs detalhados são tarefas que muitas vezes não estão disponíveis para todas as equipes de desenvolvimento. A falta de profissionais pode resultar em problemas de qualidade ou problemas na implementação de funcionalidades.
* **Foco no Detalhe:** Algumas situações podem exigir um nível excessivo de detalhe para cada funcionalidade, o que pode resultar em documentação excessiva. O detalhamento pode aumentar o tempo e o trabalho necessários para o desenvolvimento, embora ajude a garantir que as funcionalidades sejam claras e implementadas corretamente. Isso pode ocasionalmente limitar a flexibilidade da equipe e dificultar a adaptação rápida às mudanças nas necessidades ou prioridades da empresa.

# FDD na prática

Para que o FDD seja implementado com sucesso, as equipes de desenvolvimento precisam adotar uma variedade de práticas e ferramentas que apoiem a metodologia. Alguns desses costumes incluem:

* **Modelagem do Domínio:** Uma visão compartilhada do sistema pode ser criada usando técnicas de modelagem do domínio. A criação de diagramas de classes, diagramas de sequência e outros artefatos visuais que ajudam a comunicar a estrutura e o comportamento do sistema pode ser um exemplo disso.
* **Planejamento Incremental:** Dividir o desenvolvimento em etapas menores, cada uma das quais se concentra na entrega de um conjunto específico de recursos. O planejamento incremental ajuda a equipe a criar valor contínuo e reagir rapidamente a mudanças.
* **Revisões de Código:** Revisões regulares do código garantem que as funcionalidades sejam executadas corretamente e que o código atenda aos padrões de qualidade. Além disso, as revisões de código ajudam a encontrar e corrigir problemas de design ou implementação antes que se tornem problemas mais graves.
* **Automação de Testes:** Garanta que as funcionalidades sejam testadas de forma consistente e eficiente usando ferramentas de automação de testes.
* **Integração Contínua:** Garanta que o código criado por vários membros da equipe seja integrado regularmente usando técnicas de integração contínua. Isso facilita a identificação e resolução precoces de conflitos de integração, aumentando a estabilidade do sistema e diminuindo a probabilidade de problemas no futuro.
* **Entrega Contínua:** Crie uma pipeline de entrega contínua para permitir a implementação automatizada das funções em laboratórios de teste e produção. A entrega contínua permite a entrega rápida de valor ao cliente e permite feedback instantâneo sobre novas funcionalidades.
* **Comunicação Efetiva:** Todos os membros da equipe e stakeholders devem manter uma comunicação aberta e eficaz. Práticas comuns, como reuniões diárias e revisões de sprint, ajudam a garantir que todos estejam alinhados e que qualquer problema ou impedimento seja encontrado e resolvido rapidamente.

# Estudo de Caso: Implementação do FDD

Consideremos um estudo de caso de uma empresa de desenvolvimento de software que implementou a metodologia para um projeto complexo para ilustrar como o FDD pode ser aplicado na prática.

- A SoftTech foi contratada para criar um sistema de gerenciamento de estoque para uma grande rede de varejo. O projeto incluiu o desenvolvimento de um sistema que permitiria aos funcionários monitorar as vendas, gerenciar o estoque em várias lojas e produzir relatórios abrangentes sobre o desempenho das lojas.

- Para começar, a equipe da SoftTech realizou vários workshops de modelagem do domínio com os principais stakeholders com o objetivo de obter uma melhor compreensão dos processos de negócios e determinar as principais entidades e conexões no sistema de gerenciamento de inventário. A equipe criou uma visão geral do modelo que serviu como base para o desenvolvimento com base nesses workshops.

- Em seguida, a equipe fez uma lista abrangente de características que precisavam ser usadas. Funcionalidades como "acrescentar um novo produto ao estoque", "atualizar a quantidade de estoque", "registrar vendas de produto" e "gerar relatório de vendas" foram listadas na lista. Cada funcionalidade foi detalhada em detalhes e sua priorização foi baseada no valor da empresa.

- A próxima etapa foi o planejamento por funcionalidade. Nessa etapa, a equipe criou um plano detalhado para o desenvolvimento e entrega de cada funcionalidade. Para garantir que o desenvolvimento progredisse de forma organizada e eficaz, foram estabelecidas responsabilidades e prazos.

- A equipe da SoftTech fez diagramas de sequência durante o processo de design e descobriu as classes e métodos necessários para implementar cada funcionalidade. O design detalhado ajudou a garantir que a implementação fosse bem executada e que todos os requisitos e dependências fossem identificados antecipadamente.

- A equipe finalmente entrou na fase de construção, onde cada característica foi desenvolvida e testada em pequenas etapas. Para garantir que o código fosse integrado regularmente e que as funcionalidades fossem testadas de forma consistente, a integração contínua e a automação de testes foram implementadas. A entrega contínua permitiu que as novas funcionalidades fossem implementadas rapidamente, o que permitiu que os usuários finais dessem feedback imediato.

- Um sistema de gerenciamento de inventário robusto e eficaz foi criado como resultado, atendendo às demandas da rede de varejo e aumentando significativamente a eficiência operacional. A técnica FDD permitiu que a equipe da SoftTech criasse valor continuamente enquanto atendia às necessidades de negócios e produzisse software de alta qualidade.

# Conclusão

A abordagem FDD é uma abordagem ágil que se concentra na entrega contínua e crescente de benefícios valiosos para o cliente. Embora apresente desafios, suas vantagens de estrutura e foco no valor o tornam uma escolha útil para muitos projetos de software. As equipes podem melhorar a eficiência e a qualidade de seus processos de desenvolvimento ao entender e implementar o FDD de forma eficaz.