FUNDAÇÃO EDUCACIONAL DE MACAÉ - FUNEMAC

FACULDADE PROFESSOR MIGUEL ÂNGELO DA SILVA SANTOS - FeMASS

MODELOS ÁGEIS DE GERENCIAMENTO DE PROJETOS

Disciplina: Gerência de Sistemas e Projetos

Junho de 2016.

METODOLOGIAS ÁGEIS DE GERENCIAMENTO DE PROJETOS:

Feature Driven Development (FDD) e Dynamic Systems Development (DSDM)

Trabalho de Metodologia Ágil para Gerenciamento de Projetos à disciplina de Gerência de Sistemas e Projetos, apresentado ao professor Isac Mendes Lacerda pelos alunos Tiago Araujo Fróese e Victor Rangel Monteiro Maia.

Feature Driven Development (FDD)

A metodologia FDD (em português conhecida como Desenvolvimento Guiado por Funcionalidades). A FDD nasceu no ano de 1997 em um grande projeto utilizando a linguagem Java para o United Overseas Bank, um banco localizado num pequeno país do sudeste asiático, chamado Singapura. Aparece da experiência de análise e modelagem orientadas por objetos de Peter Coad, e de gerenciamento de projetos de Jeff De Luca. Inicialmente teve sua publicação no ano de 1999, no capítulo 6 do livro "Java Modeling in Color with UML", de Peter Coad, Eric Lefebvre e Jeff De Luca. Já no século XXI, no ano de 2002, Stephen Palmer (gerente de desenvolvimento do projeto em Singapura) e John Mac Felsing (arquiteto senior na TogetherSoft) publicaram o livro "A PracticalGuideto Feature Driven Development", com a versão completa, atualizada e comentada da metodologia.

A metodologia ágil FDD (Feature Driven Development), é utilizada no gerenciamento e desenvolvimento de softwares. Essa metodologia combina as melhores práticas do gerenciamento ágil de projetos com uma abordagem completa para Engenharia de Software orientada por objetos, conquistando os três principais públicos de um projeto de software: clientes, gerentes e desenvolvedores.

É uma metodologia que oferece estrutura para equipes pequenas e medias, enfatiza a produção de softwares de qualidade e funcionais. Realiza um trabalho significativo desde o início do projeto, antes de tornar-se altamente iterativa. Além disto, fornece informações de estado e progresso de forma simples e compreensível, agradando assim aos gerentes e desenvolvedores.

É uma metodologia guiada por funcionalidades, onde as mesmas devem agregar valor para o cliente e possa ser desenvolvida em, no máximo, duas semanas. Para os mais experientes, uma funcionalidade pode ser comparada a um requisito funcional. O template de uma funcionalidade é: [ação] [resultado] [objeto]. Alguns exemplos de funcionalidades: Calcular o desconto de uma venda, Validar o CPF de um cliente, Ordenar por Cidade e UF o relatório de clientes.

Características da Metodologia FDD:

- Resultados úteis a cada duas semanas ou menos.
- Blocos bem pequenos de funcionalidade valorizada pelo cliente, chamados "Features".
- Planejamento detalhado e guia para medição.
- Rastreabilidade e relatórios com incrível precisão.
- Monitoramento detalhado dentro do projeto, com resumos de alto nível para clientes e gerentes, tudo em termos de negócio.

- Fornece uma forma de saber, dentro dos primeiros 10% de um projeto, se o plano e a estimativa são sólidos.

Papéis no Feature Driven Development (FDD)

Os papeis do FDD, se dividem da seguinte forma:

- **Gerente de projeto** – É o responsável por todos os assuntos administrativos do projeto, o que inclui o gerenciamento de recursos, orçamento, equipamentos e outros. A sua principal meta é fornecer subsídios para que nenhum fator externo atrapalhe produtividade da equipe do projeto.

- **Arquiteto chefe** - Elabora o projeto geral do software a ser desenvolvido, tomando decisões finais em relação ao projeto técnico. Essa função poderá ser dividida entre o Projetista de Domínio e o Projetista Técnico.

- **Gerente de desenvolvimento** - Responsável por gerenciar as atividades diárias do projeto resolvendo problemas que poderão ocorrer com a equipe. Pode combinar as atividades desenvolvidas pelo Arquiteto Principal e Gerente de Projeto.

- **Programador chefe** - É responsável por liderar pequenos grupos de desenvolvedores durante a construção das funcionalidades do produto. Também atua como desenvolvedor e, normalmente, é responsável pelas classes mais complexas do sistema. Possui ainda, papel fundamental nas fases de absorção do conhecimento de negócio e no planejamento das funcionalidades.

- **Proprietário de classe** - Geralmente é o programador com maior experiência dentro da equipe, participando na análise dos requisitos e no projeto do software. Considerado como um dos papéis mais importantes no projeto FDD. Atua principalmente nas duas últimas etapas do processo.

- **Especialista do domínio** - Pode ser um usuário, analista de negócios, cliente ou qualquer pessoa que conheça bem o domínio do problema. Sua tarefa é informar as funcionalidades que deverão ser atendidas pelo software e entender como os requisitos estão sendo desenvolvidos.

Observação: Na metodologia FDD em projetos maiores pode existir a necessidade, outros papéis podem ser incluídos no projeto, tais como: Gerente de Versão; Guru da Linguagem; Criador de Ferramentas; Testadores e Implantadores.

Práticas no Feature Driven Development (FDD)

As práticas da metodologia FDD se divide da seguinte maneira:

- **Modelagem de objetos do domínio (negócio)** - Construção de diagramas de classes UML (Unified Modeling Language) que descrevem os objetos relevantes dentro do domínio do problema, bem como os relacionamentos entre eles. Para complementar os diagramas de classe UML, são desenvolvidos diagramas de sequência UML que descrevem explicitamente como os objetos interagem para cumprir suas responsabilidades.

- **Desenvolvimento por funcionalidade** - É feita a identificação das funcionalidades do sistema (definidas pelo cliente). Após isso, inicia-se o projeto e a construção de cada uma delas. Uma vez identificadas, as funcionalidades serão utilizadas para guiar o desenvolvimento no FDD, tendo como objetivo mostrar o progresso através da implementação das mesmas. A execução das funcionalidades, ou conjunto delas, não deve exceder de duas semanas.

- **Posse individual de classe (código)** - Cada classe ou conjunto de classes é de responsabilidade de um indivíduo. Isso pode ser uma vantagem e uma grande desvantagem em alguns pontos de vista, pois a saída do programador proprietário da classe pode gerar perda de tempo no projeto.

- **Time de funcionalidades** - A prática da propriedade individual da classe atribui classes a desenvolvedores específicos. Contudo, sabe se que o desenvolvimento deve ser por funcionalidade. Por isso, são definidas equipes, com seus respectivos desenvolvedores líderes, onde os componentes possuem as propriedades das classes, e é atribuído a eles um conjunto de funcionalidades. A equipe de funcionalidades deve ter no mínimo 3 e no máximo 6 programadores envolvidos.

- **Inspeções de modelo e de código** - Devem ser feitas durante e ao final de cada iteração, para assegurar a qualidade do projeto e do código. O objetivo principal das inspeções é a detecção de defeitos. É uma ferramenta para eliminação de erros e uma grande oportunidade de aprendizado.

- **Builds regulares** - Devem ocorrer durante as iterações, na execução de um conjunto de funcionalidades, para detectar, prematuramente, erros de integração. Uma construção regular assegura também que haja sempre um sistema atual e executável para ser apresentado ao cliente.

- **Gerenciamento de configuração** - Utilização de um sistema de controle de versões para datar e manter um histórico das alterações feitas em cada classe. Bem como, no que se refere aos requisitos, análise e o projeto de modo que facilite a visualização das modificações feitas.

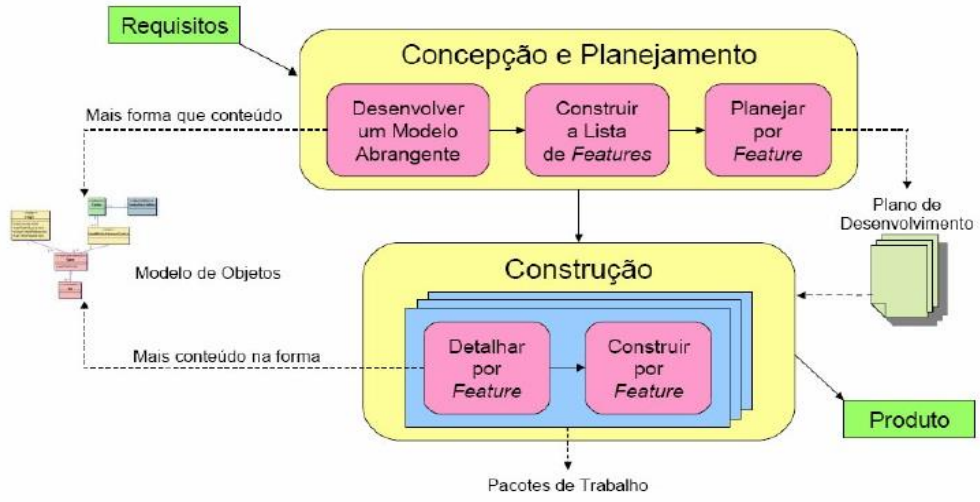
- **Relatório/visibilidade de resultados** - O FDD sugere que todos os resultados ocorridos durante o projeto sejam disseminados para todos os membros da equipe e clientes.

Fluxo/Artefatos do (FDD)

O fluxo na metodologia FDD e composta por fases e processos:

As fases:

- Concepção & Planejamento: Pensar um pouco antes de fazer (entre 1 e 2 semanas).
- Construção: Fazer de forma iterativa (geralmente em interações de 2 semanas).



Os processos:

- **Desenvolver Um Modelo Abrangente (DMA):** Análise Orientada Por Objetos. Pode envolver desenvolvimento de requisitos, análise orientada por objetos, modelagem lógica de dados e outras técnicas para entendimento do domínio de negócio em questão.

Artefatos Produzidos: Diagramas de classes, sequência, atividades, casos de uso, lista preliminar de requisitos. Anotações nos modelos Processo.

- **Construir A Lista De Funcionalidades (CLF):** Decomposição Funcional. Decomposição funcional do modelo do domínio, em três camadas típicas: áreas de negócio, atividades de negócio e passos automatizados da atividade (funcionalidades).

  Artefatos Produzidos: Lista de Funcionalidades, requisitos mais detalhados Processo.

- **Planejar Por Funcionalidade (PPF):** Planejamento Incremental. Abrange a estimativa de complexidade e dependência das funcionalidades, também levando em consideração a prioridade e valor para o negócio/cliente.

  Artefatos Produzidos: Plano de desenvolvimento, pacotes de trabalho, lista de classes com seus donos Processo.

- **Detalhar Por Funcionalidade (DPF):** Projeto Orientado Por Objetos.
  Já dentro de uma iteração de construção, a equipe detalha os requisitos e outros artefatos para a codificação de cada funcionalidade, incluindo os testes. O projeto para as funcionalidades é inspecionado.

  Artefatos Produzidos: Modelos detalhados (classes e sequência), Esqueletos de classes com métodos, pacote de trabalho detalhado, Relatório de inspeção do design, relatório de progresso atualizado Processo.

- **Construir Por Funcionalidade (CPF):** Programação e Teste Orientado Por Objetos. Cada esqueleto de código é preenchido, testado e inspecionado.

  Artefatos Produzidos: Código fonte testado e integrado, relatórios de inspeção e testes, lista de alterações feitas/necessárias, relatório de progresso atualizado observação;

Dynamic Systems Development (DSDM)

Dynamic Systems Development é um modelo de desenvolvimento de software ágil, criado na década de 90 na Inglaterra. A ideia da constituição do método inicia-se por um consorcio de vendedores e desenvolvedores, compartilhando entre si ideias e experiências de mercado em relação produção de software. Logo a ferramenta DSDM, visa solucionar problemas de construção de software como: a falta de tempo, orçamentos apertados ou até mesmo a falta de envolvimento entre membros da equipe de produção.

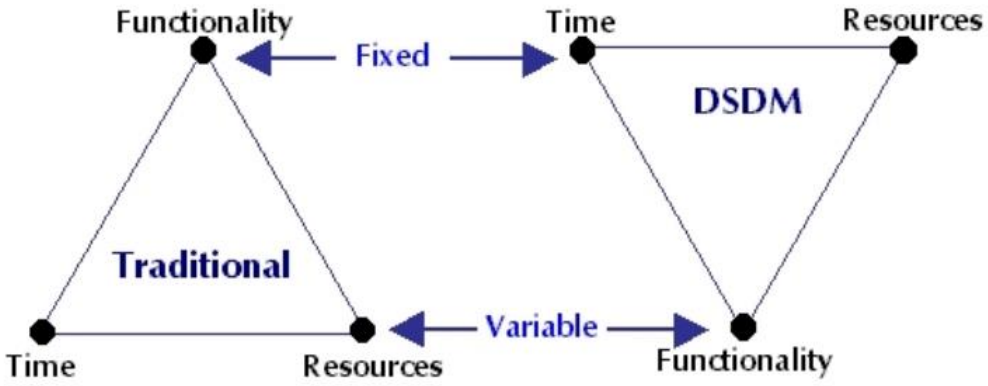


Fig.1. *Princípio fundamental da DSDM*

Papéis de Dynamic Systems Development (DSDM)

Os papeis do DSDM, se dividem em três grupos:

Primeiro Grupo:

- ***Gerente Executivo:*** Tem a função em cumprir prazos, onde a sua palavra é grande importância na tomada de decisões.
- ***Visionário:*** É o que inicia o projeto certificando que os requisitos foram bem estruturados. Ele deve conter a percepção dos objetivos de negócio projeto.
- ***Intermediador:*** É um profissional que agrega conhecimento de outras áreas para a construção do projeto, fornecendo uma quantidade feedback de usuários a equipe de desenvolvedores.

- ***Anunciante:*** Usuário que dispõe seu ponto de vista em relação ao projeto e agregando conhecimento ao projeto.

Segundo Grupo:

- ***Gerente de Projeto:*** Profissional responsável por gerenciar o projeto como um todo.
- ***Coordenador Técnico:*** Responsável pela arquitetura do sistema e controle da qualidade técnica do projeto.
- ***Líder de Equipe:*** Responsável por manter a harmonia do projeto e a realização das atividades em grupo.
- ***Desenvolvedor:*** Interpreta o modelo e requisitos do sistema incluindo desenvolvimento de artefatos de código e construção de protótipos.

Terceiro Grupo:

- ***Testador:*** Confere a parte técnica através da execução de ferramentas de teste das funcionalidades desenvolvidas, realizando comentários e documentações.
- ***Escrivão:*** Recolhe e armazenam requisitos, acordos realizados entre a equipe de trabalho.
- ***Facilitador:*** Coordena as reuniões e inspeções, sendo peça fundamental na comunicação.

Práticas de Dynamic Systems Development (DSDM)

As práticas da metodologia DSDM se dividem da seguinte maneira:

- A entrega do produto deve aproximar-se da necessidade do negócio.

- Nenhum Sistema é concluído na primeira tentativa. O processo de desenvolvimento baseia-se no princípio de Pareto onde 80% da solução de software pode ser desenvolvida em 20% do tempo empregado para encontrar a solução perfeita.

- A entrega deve ser feita na data estipulada, dentro do orçamento previsto e com boa qualidade.

- Cada etapa de desenvolvimento deve ser completada até que seja possível iniciar um novo passo. Logo cada fase do projeto pode iniciar sem depender do fim da anterior.

- Boa comunicação e envolvimento, seja ele por parte do stakeholder, usuário e dos membros referente a equipe do projeto.

- Testes integrados ao ciclo de vida

Fluxo do (DSDM)

No Dynamic Systems Development, a fase se divide em três etapas: pré-projeto, ciclo de vida e pós-projeto.

Pré- Projeto: Nesta etapa são identificados os projetos candidatos, há a definição de orçamento e fechamento de contrato. É de grande importância essa etapa, pois nela são tratadas questões iniciais, podendo evitar problemas nas fases mais avançadas.

Ciclo de Vida: É a mais complexa, sendo composta por 5 estágios:

- ***Estudo de viabilidade:*** É realizada a avaliação se o projeto pode ser utilizar a metodologia DSDM, com os recursos e tempo disponível. Ao final é gerado um relatório, protótipo de viabilidade e registros de riscos.

- ***Estudo do negócio:*** Nesse estágio é realizado workshops, onde os envolvidos com o projeto se unem para discutir a construção do sistema.

- ***Modelo funcional:*** Os requisitos anteriores são transformados em modelos funcionais, desenvolvendo deste modo um protótipo, que é revisado por diferentes grupos. Nesse período é necessário entregar ao cliente o modelo funcional e o protótipo funcional que represente as funcionalidades referentes a iteração, com intuito de serem testadas por usuários.

- ***Projeto e construção:*** Nesse estágio a integração dos componentes funcionais deve satisfazer os anseios do usuário. Para isso se divide em sub níveis como: protótipo de desenho, calendário de tarefas, criação de protótipo de desenho e rever protótipo de desenho. Ao fim destas etapas será entregue ao cliente o protótipo de desenho para que eles efetuem os testes ao produto-protótipo.

- ***Implementação:*** Nesse estágio o sistema será testado e sua documentação será entregue aos utilizadores finais que deverão começar a ser treinados para utilizar o sistema. Nesse estágio será entregue, o sistema instalado, treinamento de usuários, documento de revisão do projeto.

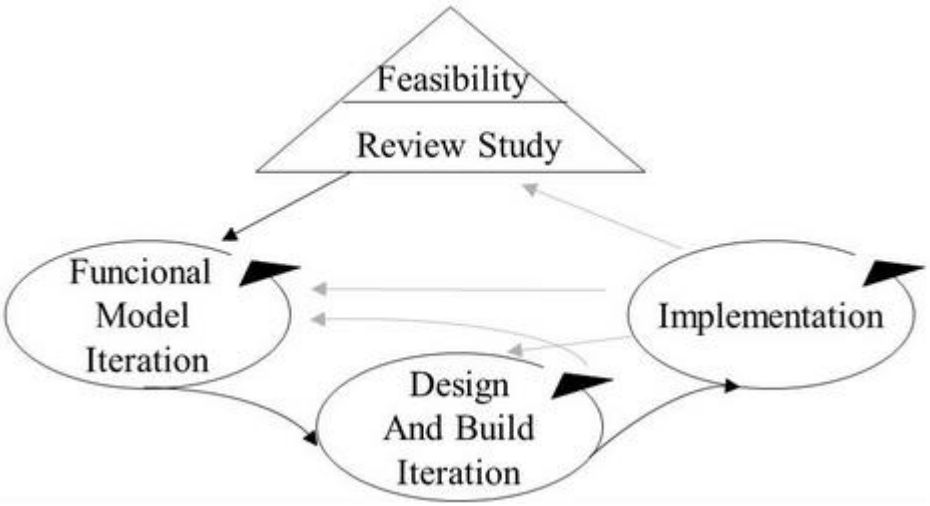


Fig.2. *Ciclo de Vida DSDM*

Pós- Projeto: Nessa etapa é garantida a eficácia do projeto, através dos processos de manutenção e melhorias de acordo com as premissas do DSDM. A manutenção é tida como continua, assim em vez de finalizar o ciclo de vida, é possível retornar fases/estágios a fim de refinar e melhorar cada vez mais a etapa concluída.

Artefatos do (DSDM)

- **Time-boxes:** Um período fixo para a execução do projeto, colocando até datas de entrega. Caso haja alguma funcionalidade que não possa ser implementada durante o período estipulado, ela deve ser feita antes da fase de pós-projeto.

- **MoSCoW:** Prioriza requisitos durante o período de desenvolvimento. A ideia fundamental é priorizar e implementar os requisitos que sejam considerados importantes, deixando o menor valor para um segundo momento.

- **Modelagem:** Processo não burocrático, com intuito de prover um melhor entendimento do problema e de uma possível solução.

- **Prototipação:** Verifica a adequação dos requisitos e facilita as discussões com o cliente. O protótipo criado deve evoluir juntamente com o projeto.

- **Teste:**Processo executado de forma sistematicamente e contínua durante o projeto.

- **Gerência de configuração:** Atividade essencial, visto que os produtos são entregues com uma grande frequência.

Análise Comparativa entre (FDD) e (DSDM)

- **Características:**

Características	FDD	DSDM
Abordagem	Interativo	Interativo
Período de interação	De 2 dias a 2 semanas	80% da solução é gerada por 20% de tempo empregado
Componentes da Equipe	Muitos membros e mais um	Tamanhos variados
Tamanho do projeto adequado	Projeto mais complexos	Todos os tipos de projetos
Envolvimento do Usuário	Envolvimento através de reports	Envolvimento através de reports
Documentação	Documentação é importante	Documentação é importante
Principais práticas	Diagramas UML	Prototipação e viabilidade de negócios

- **Vantagens:**

Feature Driven Development (FDD)	**Dynamic Systems Development (DSDM)**
Recomendado para qualquer tipo de desenvolvimento	Produz resultados
Foco em “características de valor para o cliente	Pode ser utilizado em projetos que necessitem constante a avaliação dos clientes
FDD prioriza aquilo que o cliente prioriza	Planejamento pode ser adaptado em qualquer fase do projeto
FDD possui requisitos mais formais	Enfatiza um ciclo de desenvolvimento curto (60/90 dias)
Seus princípios e práticas proporcionam um equilíbrio entre as filosofias tradicionais e as mais extremas, proporcionando uma transição mais suave para organizações mais conservadoras	Visibilidade mais cedo (protótipo)

- **Desvantagens**

Ainda existe questionamento sobre a eficácia / aplicabilidade de FDD	Sucessos anteriores são difíceis de reproduzir
Controvérsias sobre o tamanho mínimo de um time FDD	O Envolvimento com o usuário tem que ser ativo

A metodologia ágil FDD, é uma ferramenta que busca o equilíbrio entre a liberdade e o controle, tendo como alvo equipes pequenas e médias, sem que com isso perca sua principal característica que é buscar uma melhor agilidade no desenvolvimento de softwares e o método DSDM visa entregar o software num curto espaço de tempo, utilizando os preceitos de Pareto, onde uma aplicação pode ser construída 80% da mesma utilizando 20% do tempo que levaria para entregar o produto. Estar preocupado no cumprimento do prazo de entrega e tendo como diferencial a capacidade de ser flexível, tendo seu forte a interação com o cliente, extraindo informações e testes.

Referência Bibliográfica

[1] Teixeira, D. - DSDM – Dynamic Systems Development Methodology – Faculdade De Engenharia da Universidade Do Porto. Acesso, 29 de maio.

[2] Siqueira, L.F – Métodos ágeis. Acesso, 30 de maio.

[3] Luz, D.G – USP – Universidade do Estado de São Paulo – Métodos ágeis. Acesso em 29 de maio.

[4] Pressman, S. R. – Engenharia de Software – Uma abordagem Profissional. Acesso, 1 de junho.

[5] FDD Numa Casca de banana, Um guia de rápido aprendizado para a Feature Driven Development; Mar 2007; Alexandre Magno Figueiredo.

[6]<http://engenheirosdosoftware.blogspot.com.br/2008/10/fdd-feature-driven-development.html> Acessado em 05/06/2016.

[7]<http://ariston-araujo.blogspot.com.br/2012/05/vbehaviorurldefaultvmlo.html> Acessado em 05/06/2016.

[8] Feature Driven Development; 2007; Setti Rodrigo, Gameiro Lucas, BoscariolLendro, Leal Flavio.