

Engenharia de Software

Seção 6 – Modelagem Ágil / FDD / SCRUM

Objetivos

O aluno deverá reconhecer os conceitos básicos de mais alguns Métodos Ágeis, em especial SCRUM.



Na aula passada...

- ✓ Contexto histórico do surgimento de métodos ágeis
- ✓ O Manifesto Ágil
- ✓ 12 Princípios do Manifesto Ágil
- ✓ O que, porque e como
- ✓ Processos Ágeis
- ✓ Modelos Ágeis
 - ✓ XP – Extreme Programming
 - ✓ ADS – Adaptive Software Development
(Desenvolvimento Adaptativo de Software)
 - ✓ DSDM - Dynamic Systems Development Method
(Método Dinâmico de Desenvolvimento de Sistemas)

Modelagem Ágil

- ✓ A Modelagem Ágil (*Agile Modeling*) é uma metodologia baseada em práticas para documentação e modelagem efetivas de sistemas baseados em software.
- ✓ Em um nível mais alto, Modelagem ágil é um conjunto de boas práticas.

Modelagem Ágil (continuação)

- ✓ Em um nível mais detalhado, é uma coleção de valores, princípios e práticas para a modelagem de software de maneira efetiva e leve.
- ✓ O foco principal é manter a modelagem, mas eliminar a burocracia.

Modelagem Ágil (continuação)

- ✓ A Modelagem Ágil pode ser utilizada, em princípio, para melhorar outros processos de desenvolvimento.
- ✓ A Modelagem Ágil é aplicada a um processo base (RUP, XP, FDD, etc.) ao qual já se adicionam outras técnicas (como SCRUM).

Modelagem Ágil (continuação)

- ✓ Os valores de Modelagem Ágil adotam e estendem os do XP:
 - ✓ **Comunicação**: entre todos os stakeholders;
 - ✓ Ter como meta sempre a **simplicidade** do design;
 - ✓ Ter **coragem** para tomar e manter decisões;
 - ✓ Ter **humildade** para admitir que se pode estar errado, que não se sabe tudo e que pode ser necessário fazer tudo de novo.

Modelagem Ágil (continuação)

A Modelagem Ágil segue a vários princípios, vários dos quais claramente aderentes aos princípios do Manifesto Ágil:

- ✓ **Por que você está modelando?** Deve haver alguma razão para fazer um modelo. Deve-se identificar o que está sendo modelado, por que e para quem, e então elaborar um modelo apenas com o nível necessário de detalhe.

Modelagem Ágil (continuação)

- ✓ **Viajar leve.** Cada artefato criado deverá ser mantido. Identificar se vale a pena manter os artefatos criados. Há um compromisso entre documentar para tornar o entendimento mais claro, e manter o artefato.
- ✓ **Múltiplos modelos.** Nem todo sistema precisa de todos os modelos possíveis.

Modelagem Ágil (continuação)

- ✓ **Assumir a simplicidade.** Assumir que a solução mais simples é a melhor solução. Não modelar hoje o que não é pedido hoje.
- ✓ **Mudanças incrementais.** Os modelos não devem estar certos logo na primeira vez. Eles provavelmente não estarão mesmo. Fazer modelos simples, de alto nível, e detalhá-los **se necessário**.

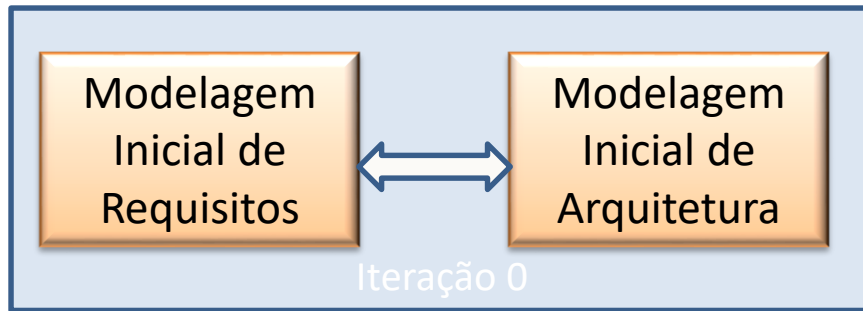
Modelagem Ágil (continuação)

- ✓ **Habilitar o próximo esforço.** É necessário pensar no próximo ciclo para o software entregue. Haverá uma nova versão? Será a mesma equipe? Qual o nível de documentação necessário para que a nova equipe seja efetiva?
- ✓ **Conteúdo é mais importante que representação.** Nem todo modelo precisa tornar-se um documento, mantido e controlado.

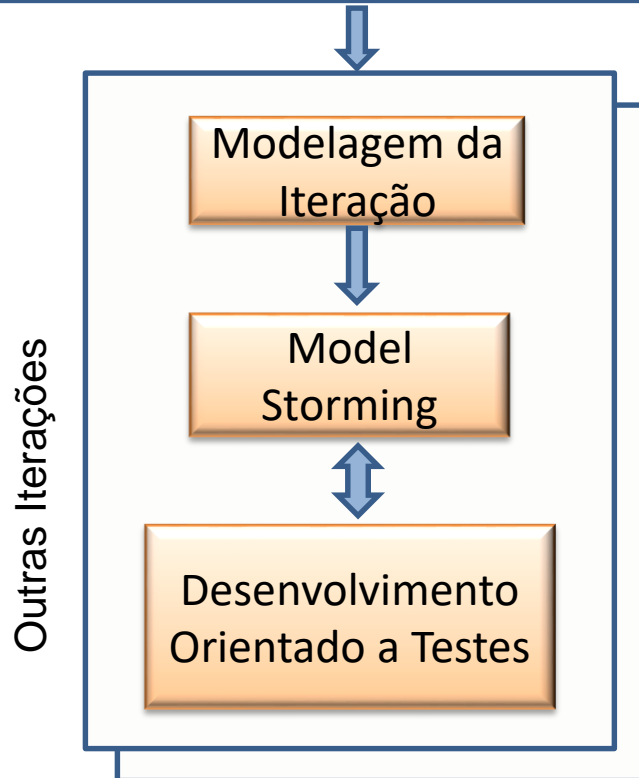
AMDD

- ✓ **AMDD** (Agile Model Driven Development – Desenvolvimento Ágil Orientado a Modelos) é a versão ágil de MDD (Model Driven Development).
- ✓ MDD é uma estratégia de desenvolvimento de software em que modelos são extensivamente construídos antes que o código seja escrito.
- ✓ A diferença para AMDD é que os modelos construídos são apenas **suficientemente bons**. (“o bom é inimigo do ótimo”).

AMDD (continuação)



- ✓ Identificar o escopo de alto nível
- ✓ Identificar requisitos iniciais
- ✓ Identificar visão de arquitetura



- ✓ Planejar o trabalho da iteração
- ✓ Modelagem para obter estimativas
- ✓ Stakeholders participam ativamente
- ✓ Modelo apenas "suficiente por enquanto"
- ✓ Desenvolver software funcionando via uma estratégia de testar primeiro
- ✓ Detalhes capturados na forma de especificações executáveis

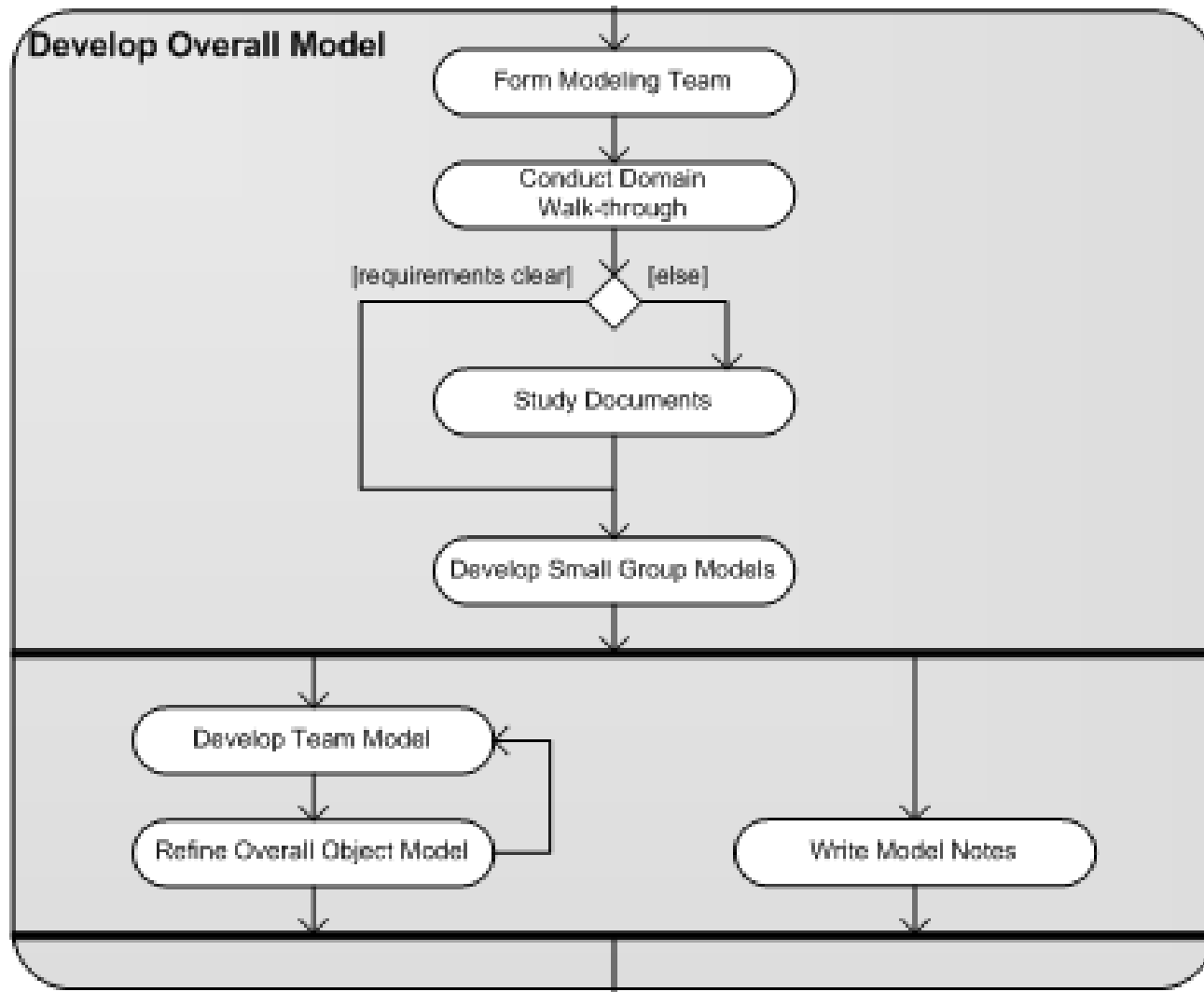
FDD

- ✓ FDD (**Feature Driven Development** – Desenvolvimento Orientado a Características) é um processo de desenvolvimento de software iterativo e incremental.


FDD (continuação)

- ✓ Foi criado originalmente por Jeff De Luca em 1997 para atender às necessidades de desenvolvimento de software de um projeto de 15 meses e 50 pessoas em um banco em Singapura.
- ✓ O processo de FDD consiste de 5 atividades básicas. Para prover um reporte de estado preciso e manter controle do desenvolvimento, são definidos marcos (milestones) que registram o progresso feito em cada característica (feature).

Criar Modelo Global



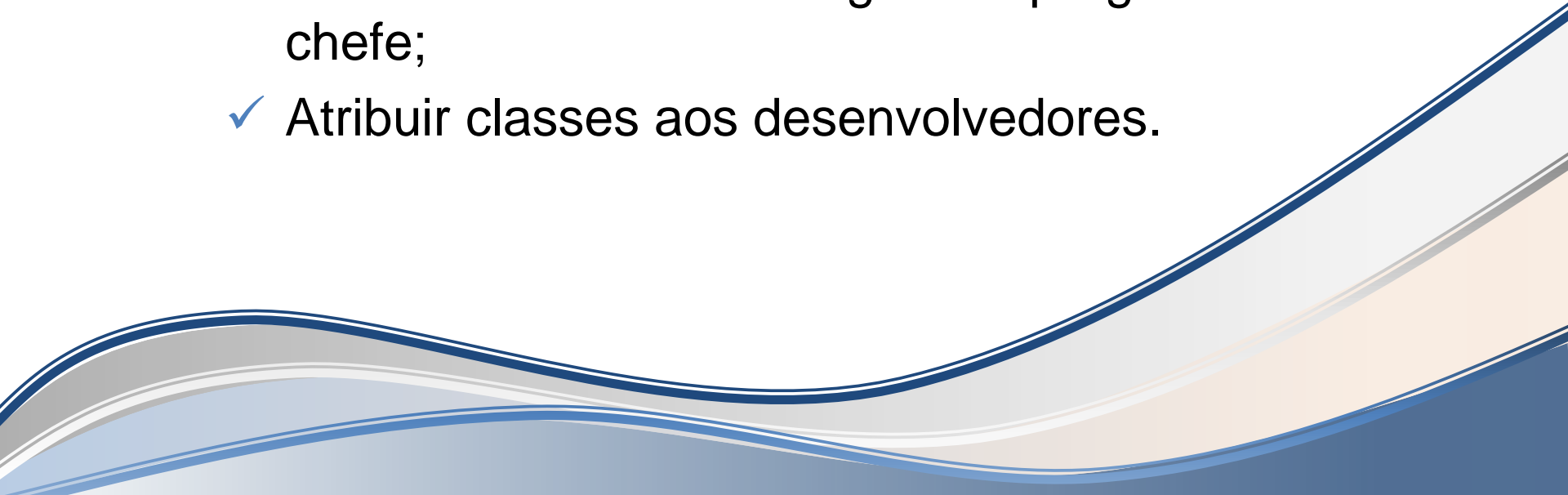
Criar Modelo Global (continuação)

- ✓ O primeiro passo é uma revisão do escopo do sistema e de seu domínio.
 - ✓ O domínio é então dividido em subdomínios, que são modelados.
 - ✓ Os modelos são revisados e são definidos os modelos finais para cada subdomínio.
 - ✓ O modelo global é definido como um agrupamento dos modelos dos subdomínios.
- 

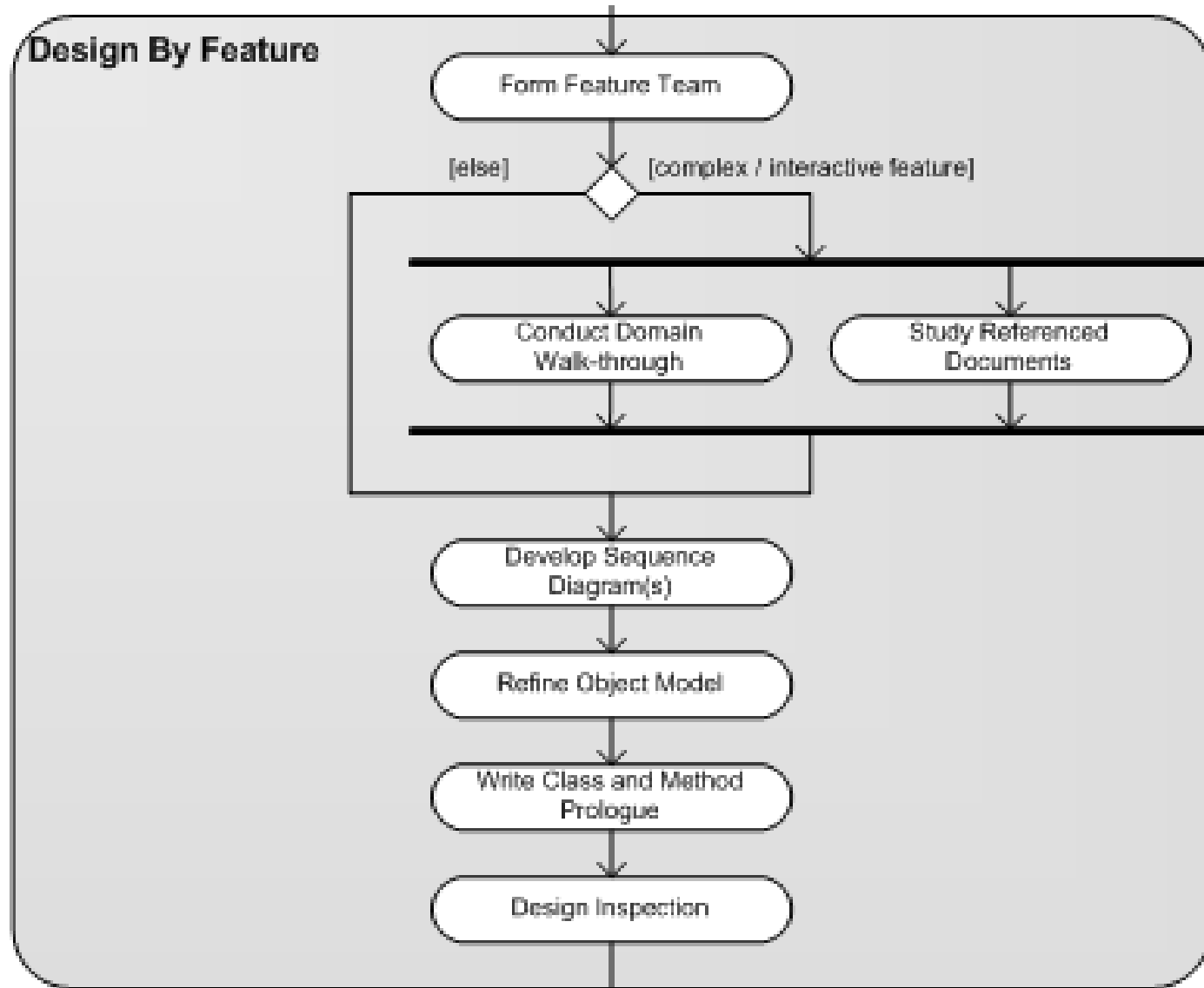
Construir Lista de Características

- ✓ Após definir a equipe, é identificado o conjunto de características (**features**) do sistema.
- ✓ O domínio e os subdomínios levantados no processo anterior são **decompostos funcionalmente**, em funções que utilizem termos que façam sentido no negócio, para o cliente.
- ✓ Features são granulares de acordo com a regra de que cada uma delas não pode levar mais de 2 semanas para ser completada.


Planejar por features

- ✓ O objetivo desta atividade é determinar a ordem em que as features serão implementadas, baseado nas suas dependências, carga sobre a equipe e complexidade. As tarefas realizadas são:
 - ✓ Determinar a sequência de desenvolvimento;
 - ✓ Atribuir atividades de negócio a programadores-chefe;
 - ✓ Atribuir classes aos desenvolvedores.
- 

Projetar por features (continuação)



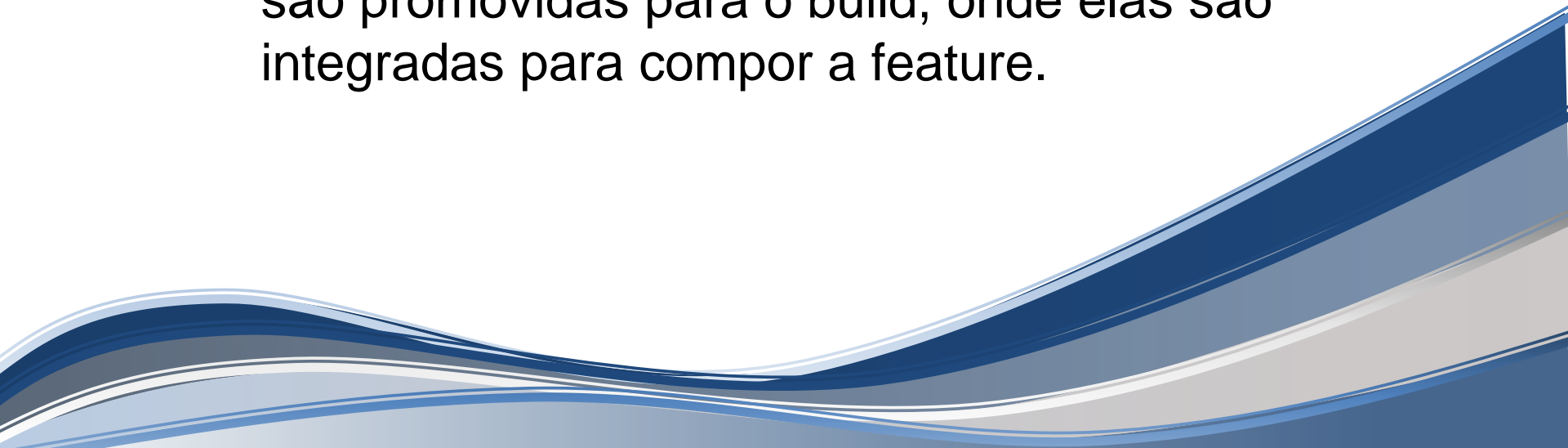
Projetar por features (continuação)

- ✓ O programador-chefe organiza a equipe com base nas classes que serão utilizadas para o desenvolvimento daquela feature.
 - ✓ São realizadas atividades de estudo das referências existentes e nas características do domínio.
- 

Projetar por features (continuação)

- ✓ São desenvolvidos diagramas de sequência que indicam como a feature será desenvolvida.
- ✓ Os modelos de classes são refinados e são desenvolvidas as classes em si e as assinaturas de métodos, exceções e parâmetros.


Implementar por feature

- ✓ As classes e métodos correspondentes à feature são completamente implementados.
 - ✓ São realizados Testes de Unidade e inspeção de código.
 - ✓ As classes completas, testadas e inspecionadas, são promovidas para o build, onde elas são integradas para compor a feature.
- 

SCRUM

- ✓ O SCRUM é um modelo ágil de processo desenvolvido por Jeff Sutherland no início da década de 90.
- ✓ O processo e a ideia inicial do nome foram baseados em uma descrição de modelo de processo criado por Hirotaka Takeuchi e Ikujiro Nonaka, segundo o qual um único grupo trabalha em todas as fases do desenvolvimento.

SCRUM (continuação)

- ✓ O SCRUM é definido por seus criadores como um “framework dentro do qual podem ser empregados vários processos e técnicas.”
 - ✓ O framework SCRUM consiste das equipes SCRUM e seus papéis associados, eventos, artefatos e regras.
 - ✓ O SCRUM é baseado em uma “teoria empírica de controle de processo”, empregando uma estratégia iterativa e incremental para otimizar a previsibilidade e reduzir riscos.
- 

A Equipe SCRUM

A eqSCRUM é composta de:

- ✓ **Gerente de Produto** (Product Owner): Responsável por maximizar o valor do produto e o resultado da Equipe de Desenvolvimento.
- ✓ **Equipe de Desenvolvimento** (Development Team): Profissionais que fazem o trabalho de disponibilizar um incremento do produto “Feito” ao final de cada ciclo de desenvolvimento.

A Equipe SCRUM

A eqSCRUM é composta de:

- ✓ **SCRUM master**: Responsável por assegurar que SCRUM é entendido e aprovado, garantindo uipe que a equipe seja aderente à teoria, prática e regras SCRUM.

Gerente de Produto

O Gerente de Produto é o único responsável por gerenciar a [Lista de Pendências do Produto](#) (Product Backlog). Este gerenciamento consiste de:

- ✓ Expressar claramente os itens da Lista de Pendências;
- ✓ Ordenar os itens da lista de modo a atingir objetivos e missões da melhor forma;

Gerente de Produto (continuação)

- ✓ Assegurar o valor do trabalho realizado pela Equipe de Desenvolvimento;
- ✓ Assegurar que a Lista de Pendências é conhecida por todos;
- ✓ Assegurar que a Equipe de Desenvolvimento entenda os itens da Lista de Pendências até o nível necessário.

Equipe de Desenvolvimento

- ✓ A equipe de desenvolvimento é auto-organizada e tem autoridade para organizar e gerenciar seu próprio trabalho.
- ✓ Equipes de Desenvolvimento possuem todas as habilidades necessárias para transformar pendências em incrementos de funcionalidades entregáveis.

Equipe de Desenvolvimento

(continuação)

- ✓ Não há títulos para os membros da equipe, a não ser “Desenvolvedor”.
- ✓ Equipes de Desenvolvimento não contêm subequipes dedicadas a tópicos específicos, como testes e análise de negócio.

SCRUM Master

- ✓ O SCRUM Master é um líder-servidor da equipe SCRUM.
- ✓ Ele realiza serviços importantes para que o trabalho sendo desenvolvido tenha poucas interrupções e gere o máximo de valor.



SCRUM Master (continuação)

- ✓ O SCRUM master auxilia aqueles fora da equipe SCRUM a entender quais de suas interações com a equipe são úteis e quais não são.
- ✓ Adicionalmente, o SCRUM master ajuda todos a modificarem suas interações de modo a maximizar o valor criado pela equipe SCRUM.

SCRUM Master (continuação)

O SCRUM master serve ao Gerente de Produto de diversas formas, entre as quais:

- ✓ Procurando técnicas para um gerenciamento efetivo da Lista de Pendências;
- ✓ Comunicando claramente visão, objetivos e itens da Lista de Pendências à Equipe de Desenvolvimento;
- ✓ Ensinando à Equipe de Desenvolvimento como criar itens claros e concisos para a Lista de Pendências;
- ✓ Facilitando os eventos SCRUM.

SCRUM Master (continuação)

O SCRUM master serve à Equipe de Desenvolvimento de diversas formas, entre as quais:

- ✓ Treinando a equipe em auto-organização e interfuncionalidade;
- ✓ Ensinando e liderando a equipe a criar produtos de alto valor agregado;
- ✓ Removendo impedimentos para o progresso da Equipe de Desenvolvimento;
- ✓ Amparando a Equipe de Desenvolvimento em uma organização onde SCRUM não esteja completamente implantado.

SCRUM Master (continuação)

O SCRUM master serve à organização de diversas formas, entre as quais:

- ✓ Liderando e treinando a organização na adoção do SCRUM;
- ✓ Planejando implementações SCRUM dentro da organização;
- ✓ Criando modificações que aumentem a produtividade da equipe SCRUM;
- ✓ Ajudando empregados e stakeholders a entender e apoiar o SCRUM e o desenvolvimento empírico de produtos.

Eventos SCRUM

- ✓ Eventos em SCRUM existem para criar regularidade e minimizar a necessidade de reuniões não previstas.
- ✓ SCRUM usa eventos de duração fixa, de forma que cada evento tenha uma duração máxima.
- ✓ Isto assegura que uma quantidade apropriada de tempo seja gasta planejando, evitando desperdício de tempo.
- ✓ Com exceção do Sprint em si, todos os eventos são oportunidades para inspeção e adaptação.

Sprint

- ✓ É o coração do SCRUM. É um período de tempo de um mês ou menos, em que um incremento de produto usável, “Pronto” e potencialmente liberável é criado.
- ✓ Um novo Sprint começa imediatamente após o término de outro Sprint.



Sprint (continuação)

- ✓ Durante o Sprint:
 - ✓ Nenhuma mudança que afete o Objetivo do Sprint é realizada;
 - ✓ A composição e os objetivos da Equipe de Desenvolvimento permanecem constantes;
 - ✓ Escopo pode ser esclarecido e renegociado entre o Gerente de Produto e a Equipe de Desenvolvimento à medida que mais é aprendido.

Sprint (continuação)

- ✓ Cada Sprint pode ser visto como um projeto de um mês de duração.
- ✓ Como tal, cada Sprint contém:
 - ✓ A definição do que deve ser construído;
 - ✓ Um plano flexível que guia a construção;
 - ✓ O trabalho a ser feito;
 - ✓ O produto resultante.
- ✓ Sprints são restritos a um mês porque, em prazos maiores, a complexidade e o risco aumentam. Assim, o risco é limitado ao custo de um mês.

Sprint (continuação)

O Sprint contém e consiste de:

- ✓ Reunião de Planejamento do Sprint;
- ✓ Scrum diário;
- ✓ O trabalho de desenvolvimento;
- ✓ Reunião de Revisão do Sprint;
- ✓ Retrospectiva do Sprint.

As Reuniões

- ✓ **Reunião de Planejamento:** Realizada no início do Sprint, define o que será realizado e como o esforço necessário será despendido.
- ✓ **Scrum diário:** Reunião de 15 minutos, diária, entre os membros da Equipe de Desenvolvimento para sincronizar atividades e criar um plano para as próximas 24 horas.



As Reuniões (continuação)

- ✓ **Revisão do Sprint:** Reunião ao final do Sprint para inspecionar o incremento e adaptar a Lista de Pendências do produto.
- ✓ **Retrospectiva do Sprint:** Reunião após a Revisão para avaliar melhorias no grupo, processo e ferramentas.

Os Artefatos

- ✓ Os artefatos do SCRUM representam trabalho ou valor em várias formas que são úteis para prover transparência e oportunidades para adaptação e inspeção.
- ✓ Os artefatos definidos pelo SCRUM são:
 - ✓ Lista de Pendências do Produto (**Product Backlog**);
 - ✓ Lista de Pendências do Sprint (**Sprint Backlog**).

Product Backlog

- ✓ O Product Backlog é a lista ordenada de tudo que poderia ser necessário no produto, e é a única fonte de requisitos para quaisquer mudanças a serem feitas no produto.
- ✓ O Product Backlog nunca está completo. Ele é dinâmico e está sempre mudando para refletir o que o produto precisa para ser útil, apropriado e competitivo.

Product Backlog (continuação)

- ✓ Ele lista todas as características, funções, requisitos, melhorias e correções que constituem as mudanças a serem feitas no produto nos releases futuros.
- ✓ O Product Backlog é frequentemente ordenado por valor, risco, prioridade e necessidade. Usualmente os itens com maiores ordens são os mais propensos a serem atacados antes.

Sprint Backlog

- ✓ O Sprint Backlog é o conjunto de itens do Product Backlog selecionados para o Sprint mais um plano para entregar o incremento do produto e realizar o Objetivo do Sprint.
- ✓ Ele é uma previsão da Equipe de Desenvolvimento sobre quais funcionalidades estarão no próximo incremento e o trabalho necessário para entregar estas funcionalidades.

Sprint Backlog (continuação)

- ✓ O Sprint Backlog é uma visão de tempo real do trabalho a ser realizado pela Equipe de Desenvolvimento, e pertence a ela. Assim, somente ela pode modificar o Sprint Backlog no decorrer do Sprint.
 - ✓ À medida que novo trabalho é requerido, ele é adicionado ao Sprint Backlog. Quando o trabalho é desempenhado ou completado, o trabalho restante é atualizado.
- 