

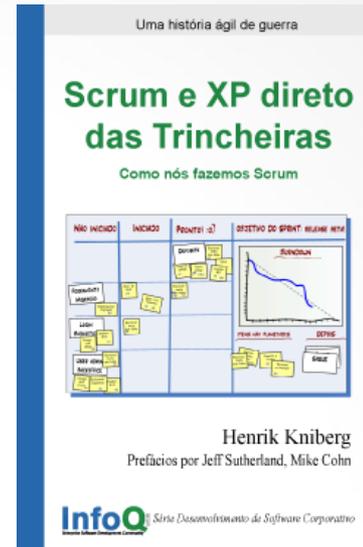
# Engenharia de Software

## Metodologias Ágeis





## Bibliografia



- Scrum e XP Direto das Trincheiras (Download gratuito disponível em <http://www.infoq.com/br/minibooks/scrum-xp-from-the-trenches> )
- Beck, Kent. [Extreme Programming Explained](#) – Embrace Change. Editora: Addison-Wesley.
- Schwaber, Ken. [Scrum Guide](#). Disponível em: <http://www.scrumguides.org/>
- [Scrum e Agile em Projetos](#) – Guia Completo – Editora: Brasport. Fábio Cruz

# Manifesto Ágil

## Manifesto para o desenvolvimento ágil de software

---

Estamos descobrindo maneiras melhores de desenvolver software fazendo-o nós mesmos e ajudando outros a fazê-lo. Através deste trabalho, passamos a valorizar:

**Indivíduos e interação entre eles** 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

Mesmo havendo valor nos itens à direita, valorizamos mais os itens à esquerda.

Kent Beck

Mike Beedle

Arie van Bennekum

Alistair Cockburn

Ward Cunningham

Martin Fowler

James Grenning

Jim Highsmith

Andrew Hunt

Ron Jeffries

Jon Kern

Brian Marick

Robert C. Martin

Steve Mellor

Ken Schwaber

Jeff Sutherland

Dave Thomas

## Princípios por trás do manifesto ágil

---

**Prioridade, satisfazer o cliente com entrega adiantada e contínua de SW de valor**

**Aceitar e adequar-se a mudanças para o cliente tirar vantagens competitivas.**

**Entregar SW funcionando com frequência, na escala de horas, dias ou semanas.**

**Pessoas de negócio e desenvolvimento trabalham em conjunto diariamente.**

**Projetos com indivíduos motivados, ambiente e suporte, fazem a diferença.**

**O Método mais eficaz de se comunicar inter e intra, é uma conversa cara a cara.**

**Software funcional é a medida primária de progresso.**

**Todos são capazes de manter passos constantes em um ambiente sustentável.**

**Contínua atenção à excelência técnica e bom design, aumenta a agilidade.**

**A arte de maximizar a quantidade de trabalho que não precisou ser feito.**

**As melhores arquiteturas e artefatos emergem de times auto-organizáveis.**

**Em intervalos regulares, o time reflete em como ficar mais efetivo e melhora.**

# Contextualização

- Alternativa aos métodos tradicionais
- Esforço para sanar fraquezas reais e perceptíveis
- Constantes mudanças da área
- Busca responder à mudanças mais rapidamente
- Baseado em abordagem iterativa e incremental
- Único artefato realmente importante é uma versão do software operacional
- Agilidade = resposta a mudanças, adaptabilidade

# Primórdios

1950 - Toyota Production System

1986 - The New New Product Development Game

1992 - Crystal Clear Method

**1993 – Scrum**

1994 - Analysis Patterns, UML Distilled, Planning XP

**1996 – XP**

1997 - DSDM (Dynamic Syst. Dev. Method)

1997 - FDD - Feature-Driven Development

1997 - ASD - Adaptive SW Develop.

1999 - The Pragmatic Programmer

2003 - Lean Software Development



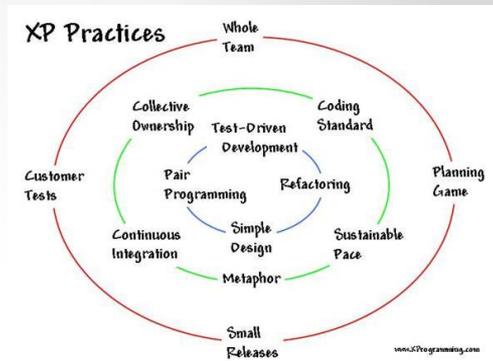
# Premissas Comuns

- Times pequenos, auto-organizados, menos hierarquia e mais autonomia;
- Foco em valor, saber o que realmente faz a diferença para o negócio e pessoas;
- Ciclos curtos, iterativos-incrementais, na escala de horas, dias e semanas;
- Busca pela qualidade consciente, ter orgulho do que se faz, sem desperdícios;
- Liberdade com responsabilidade coletiva, devemos ter orgulho de toda a equipe;
- Gestão visual, realismo e transparência, tomando decisões diariamente;
- Usar uma linguagem comum junto a todos os envolvidos no projeto ou sistema.

# XP – Extreme Programming

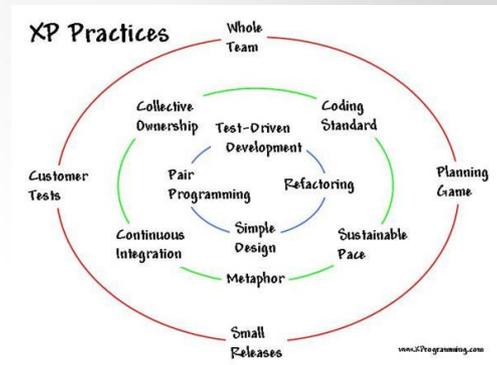
- “Trata-se de uma metodologia ágil para equipes pequenas e médias desenvolvendo software com requisitos vagos e em constante mudança” – Kent Beck
- Premissa mais famosa é que “o custo da mudança não deve aumentar dramaticamente com o passar do tempo”
- Programar ao Extremo:
  - Levar todas as boas práticas ao extremo.
    - Se testar é bom, vamos testar toda hora (testes funcionais)
    - Se revisar código é bom, vamos revisá-lo toda hora (pair programming)
    - Se projetar é bom, vamos fazer disso parte do trabalho diário de cada pessoa (refactoring)
    - Se integrar é bom, vamos integrar a maior quantidade de vezes possível (integração contínua)
    - Se simplicidade é bom, vamos deixar o sistema mais simples possível (simplicidade)

# Práticas do XP



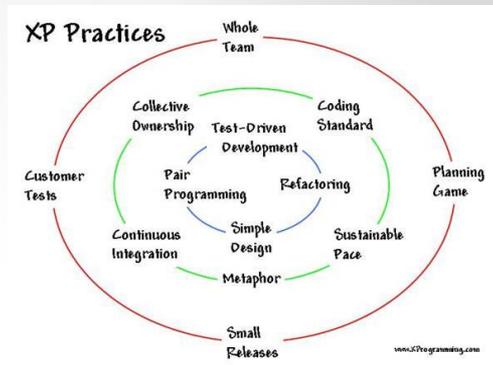
- The Customer is Always Available (O cliente sempre disponível)
  - Cliente com conhecimento, em tempo integral, disponível para colaborar em dúvidas, alterações e prioridades em um escopo.
- Metaphor (Uso de Metáforas no Projeto)
  - Como forma de facilitar a comunicação da equipe, o estabelecimento de metáforas em pontos chave do projeto permite uma fácil assimilação.
- Planning Game (Jogo do Planejamento)
  - Reuniões entre o cliente e os técnicos são estimuladas com o objetivo de definir as “users stories”, além de auxiliar a estimar o tempo ideal e a prioridade.
- Small Releases (Pequenas Versões)
  - Pequenas versões sendo entregues constantemente, como forma de detectar necessidades/feedback

# Práticas do XP



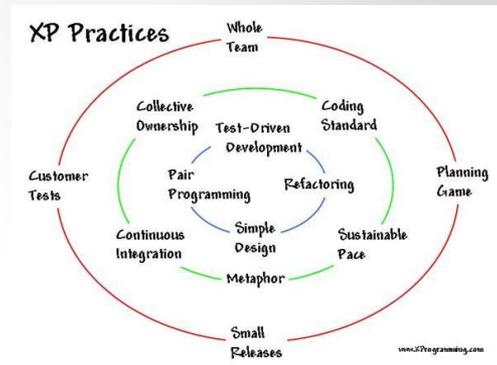
- **Acceptance Tests (Testes de Aceitação)**
  - São definidos na fase inicial os critérios para aceitação do software como forma de controlar o que está sendo entregue.
- **Test First Design (Primeiro os Testes)**
  - Garante a redução de erros de programação e aumenta a garantia de qualidade.
- **Continuous Integration (Integração Contínua)**
  - Os diversos módulos devem ser integrados tão logo sejam construídos. O código precisa obter sucesso em uma série de fatores pré-definidos.
- **Simple Design (Simplicidade de Projeto)**
  - Tudo deve ser feito de forma mais simples possível, facilitando a compreensão e possível continuidade por qualquer um dos membros.

# Práticas do XP



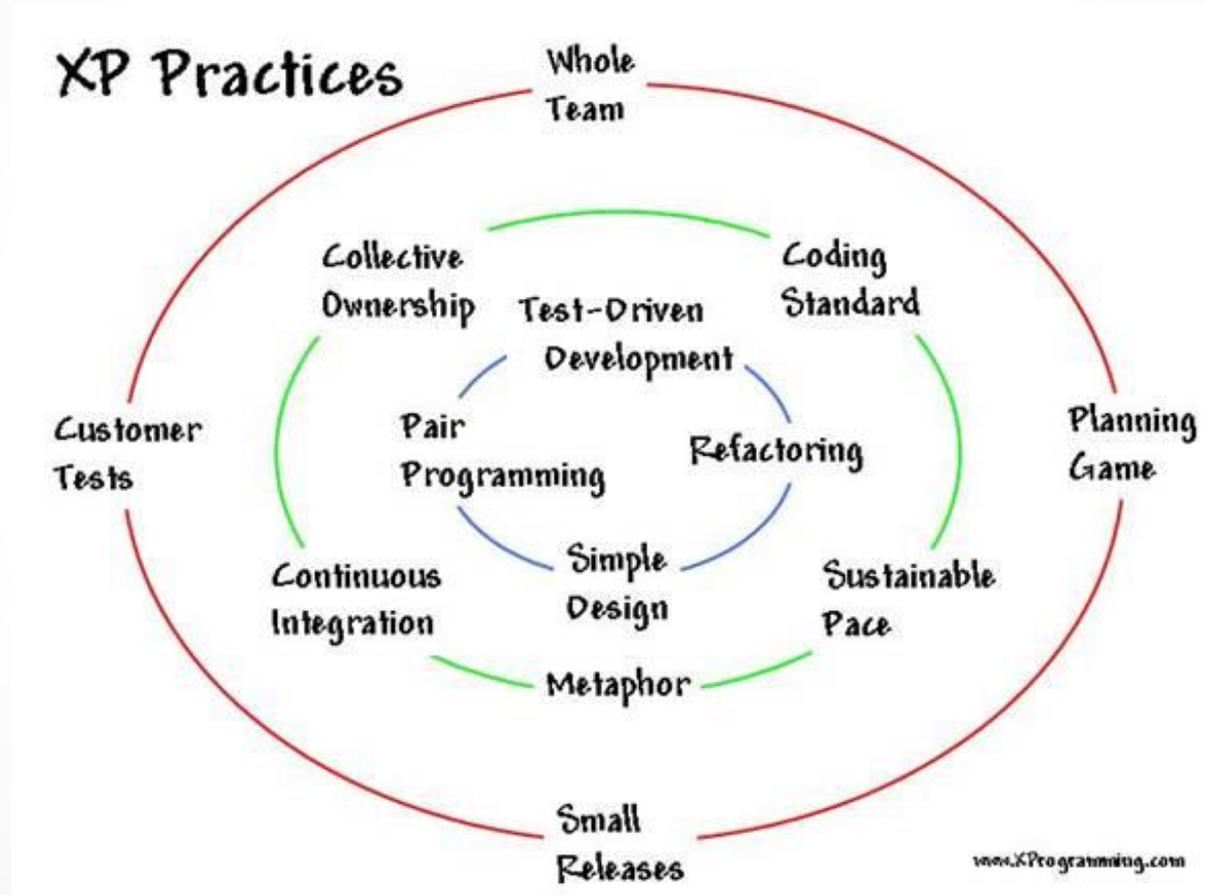
- Refactoring (Refatoração)
  - Modificação da estrutura interna do código com o objetivo de organizar ou estruturar as informações sem afetar seu compartimento externo.
- Pair Programming (Programação em Pares)
  - Intuito de elevar os níveis de atenção ao código produzido, pois um acompanha o trabalho do outro, minimizando a possibilidade de erros no código.
- Collective Code Ownership (Propriedade Coletiva do Código)
  - Todos são responsáveis pelo código, não é necessário autorização para alterar qualquer arquivo.
- Coding Standards (Padronização do Código)
  - Todo código é desenvolvido segundo um padrão, de forma que todos tenham a mesma visão do código.

# Práticas do XP



- **Sustainable Pace (Ritmo Sustentável)**
  - Recomenda a doção de semanas com 40 horas de trabalho, que possam ser mantidos em um bom ritmo durante muito tempo.
- **Whole Team (Time Coeso)**
  - Todos trabalham em conjunto promovendo a colaboração e sem hierarquias.
- **Stand Up Meetings (Reuniões em Pé)**
  - Reuniões diárias com intuito de promover o acompanhamento das tarefas do projeto.

# Práticas do XP



# Valores do XP



Imagen elaborada por [oficinaproyectosinformatica.blogspot.com](http://oficinaproyectosinformatica.blogspot.com)

- **Comunicação**
  - Providenciar a informação necessária quando necessário
- **Simplicidade**
  - Reduz a complexidade do sistema, faz apenas o que for necessário
- **Feedback**
  - Feedback sobre a qualidade do código, do usuário final, da equipe
- **Coragem**
  - Tomar as decisões corretas mesmo quando difíceis, aceitar mudanças
- **Respeito**
  - Da equipe, do cliente, do usuário, de todos os envolvidos

# Princípios do XP

- Feedback Rápido
- Presumir Simplicidade
- Mudanças Incrementais
- Abraçar as Mudanças
- Trabalho de Alta Qualidade





## 1 - CESPE – 2015 – TJ/DFT

Na metodologia XP (extreme programming), em que todos os requisitos são expressos como cenários, deve-se aguardar, após a conclusão das tarefas, ciclos de cento e oitenta dias para a publicação de grandes releases do software.(Certo/Errado)

## 2 – CESPE – 2015 – STJ

Na Extreme Programming, a programação em pares cria ilhas de especialistas na equipe por meio da análise simultânea de duas pessoas no desenvolvimento do software.(Certo/Errado)

## 3 – CESPE – 2010 – TRE/BA

Práticas de desenvolvimento de *software* aos pares de programadores, em que um programador verifica o trabalho do outro, são uma característica do método de desenvolvimento XP.(Certo/Errado)

## 1 - CESPE – 2015 – TJ/DFT

Na metodologia XP (extreme programming), em que todos os requisitos são expressos como cenários, deve-se aguardar, após a conclusão das tarefas, ciclos de cento e oitenta dias para a publicação de grandes releases do software.(Certo/Errado)

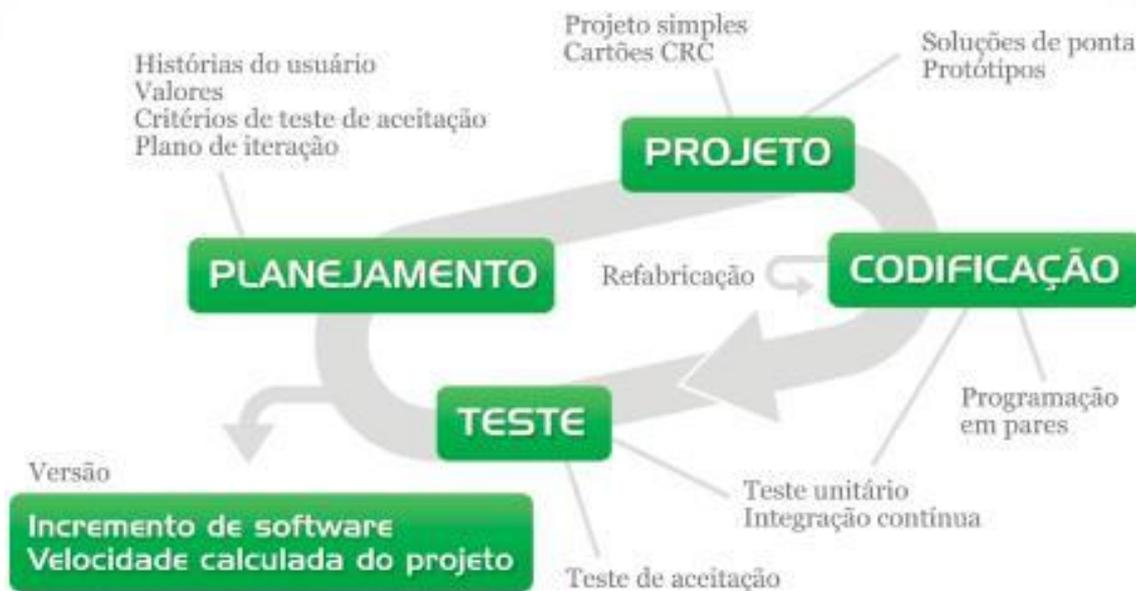
## 2 – CESPE – 2015 – STJ

Na Extreme Programming, a programação em pares cria ilhas de especialistas na equipe por meio da análise simultânea de duas pessoas no desenvolvimento do software.(Certo/Errado)

## 3 – CESPE – 2010 – TRE/BA

Práticas de desenvolvimento de *software* aos pares de programadores, em que um programador verifica o trabalho do outro, são uma característica do método de desenvolvimento XP.(Certo/Errado)

# Atividades Arcabouço XP



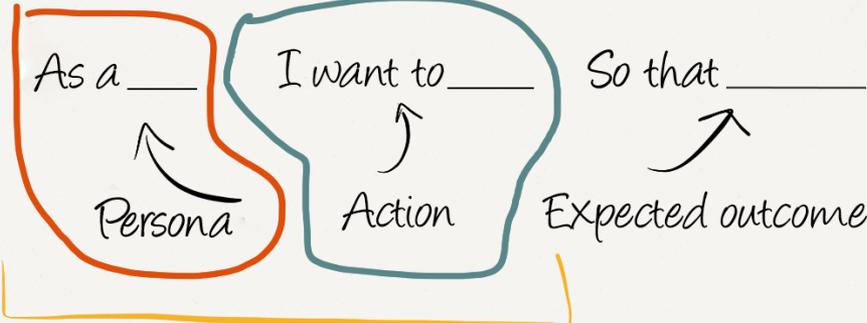
# Processo XP

- Planejamento
  - “Ouvir” -> Histórias do Usuário.
  - Cada História possui um valor atribuído pelo cliente e um custo atribuído pela equipe XP (medido em semanas)
  - Histórias com mais de 3 semanas devem ser divididas pelo cliente.
  - Clientes e equipe trabalham juntos para decidir quais histórias serão agrupadas para a versão seguinte, baseadas no “valor”, “risco” e “velocidade”.
  - Depois do primeiro incremento ter sido entregue, a equipe calcula a velocidade do projeto ( quantidade de histórias implementadas)

Irrelevant



How do we know this is the best action to take? Maybe there's a better way...



Too many assumptions

Como um [ator] eu quero/preciso de /devo/gostaria de [ação] para [funcionalidade].

Como um cliente eu quero ver os filmes disponíveis para locação para que eu possa agendar uma reserva na data X.

Como um cliente eu quero ver os filmes disponíveis para locação para que eu possa alugá-lo.

# Processo XP

- Projeto
  - KIS (Keep it Simple)
  - Guia de implementação da história da forma que foi escrita.
  - XP recomenda a criação de protótipo sempre que um difícil problema for encontrado.
  - Único artefato produzido durante o processo são cartões CRC

# Processo XP

- Codificação
  - Começa com o desenvolvimento de uma série de testes de unidades.
  - Trabalho ocorre sempre em dupla.
  - Código é integrado conforme vai sendo desenvolvido.
- Testes
  - Testes devem ser automatizados para serem executados fácil e repetidamente, encorajando a estratégia de testes de regressão.
  - Testes de integração e de sistema podem ocorrer diariamente para indicar o progresso.
  - Testes de aceitação são feitos com base nas histórias de usuário.

## 4 – CESPE – 2014 – ANTAQ

No XP (*Extreme Programming*), todos os desenvolvedores da equipe devem possuir autorização para modificar, consertar ou refatorar partes do sistema.(Certo/Errado)

## 5 – CESPE – 2014 – ANATEL

No contexto de um processo ágil, tal como o XP, é necessária a criação dos casos de usos da linguagem de modelagem unificada (UML) depois da modelagem das histórias de usuários.(Certo/Errado)

## 4 – CESPE – 2014 – ANTAQ

No XP (*Extreme Programming*), todos os desenvolvedores da equipe devem possuir autorização para modificar, consertar ou refatorar partes do sistema. (Certo/Errado)

## 5 – CESPE – 2014 – ANATEL

No contexto de um processo ágil, tal como o XP, é necessária a criação dos casos de usos da linguagem de modelagem unificada (UML) depois da modelagem das histórias de usuários. (Certo/Errado)

# Scrum

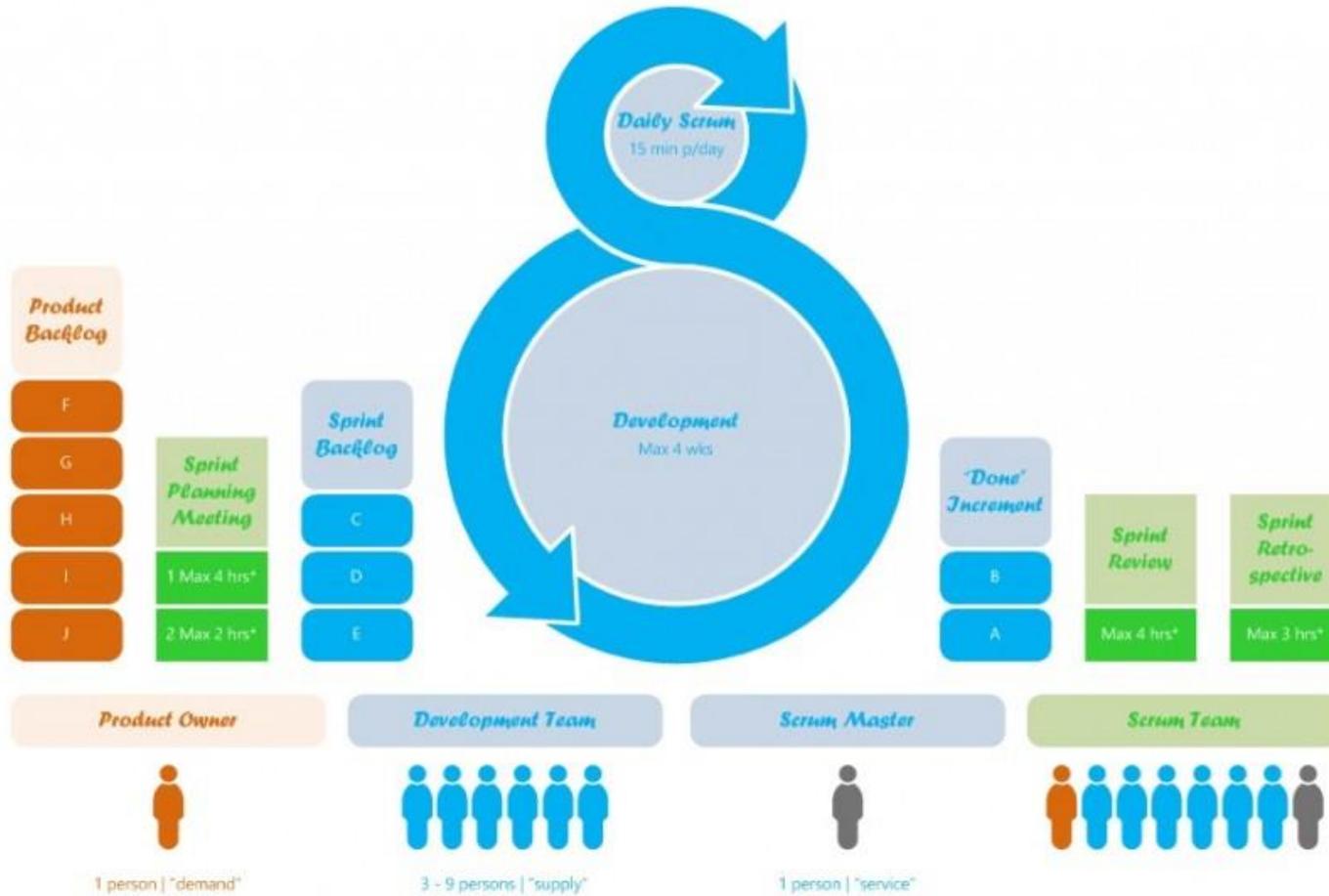


- Foco voltado para a gestão do desenvolvimento iterativo do que em atividades técnicas específicas;
- Framework de processo ágil que pode ser empregado diversos processos e técnicas
- Podem ser adicionados novos elementos de acordo com a necessidade
- Não cita nenhuma prática específica

# Características - Scrum

- Equipes auto organizadas
  - Maximizar a comunicação e diminuir a supervisão
- Iteração no Scrum é chamada de *Sprint*
  - Trabalho que se realiza durante um período de tempo para produzir um produto de trabalho
- Product Backlog
  - Lista ordenada por prioridade dos requisitos a serem implementadas pelo produto
- Framework essencialmente gerencial
- Sprint Backlog
  - Lista de tarefas, derivado do product backlog

# Scrum Overview



\* Duration of this event depends on the duration of the Sprint.

Designed by Mark Hoogveld © 2012

# Papéis

- Product Owner
  - Define as funcionalidades do produto
  - Mantém os itens do backlog atualizados e priorizados
  - Aceita ou rejeita o que foi produzido
  - Sempre disponível para esclarecer dúvidas
- Scrum Master
  - Facilitador do Projeto
  - Responsável por resolver os impedimentos do time
  - Proteger o time de interferências externas
  - Garantir que o processo Scrum foi entendido e está sendo seguido por todos
- Team
  - Multidisciplinar
  - Dedicção Integral
  - Trocas só na mudança de Sprints



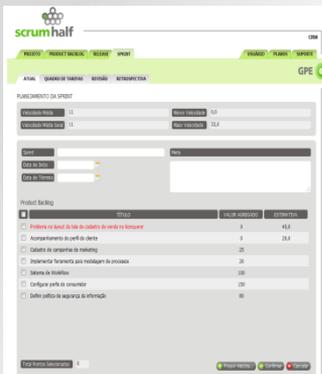
# Cerimônias

- Planejamento da Sprint
  - Selecionam-se funcionalidades do product backlog e identificam-se as tarefas;
  - Reunião dura 8 horas em duas partes:
    - 1ª parte o PO define a prioridade, seleção dos itens e meta da Sprint;
    - 2ª parte a Equipe definirá a Sprint Backlog
- Reuniões Diárias
  - Apenas os membros da equipe participam;
  - Não é para solução de problemas
  - Máximo de 15 minutos
- Revisão da Sprint
  - Apresenta os resultados que foram obtidos;
  - Apresentação é feita no formato de uma demo;
  - Sem slides, com 2 horas de preparação no máximo;
  - Pode gerar modificações no Product Backlog
- Retrospectiva da Sprint
  - Após a revisão da sprint;
  - Revisa o processo que foi seguido;



## Reunião Diária

- Trabalho é inspecionado diariamente pela própria equipe de desenvolvimento com o objetivo de trabalhar o autogerenciamento da equipe;
- Duração curta (cerca de 15min), em pé.
- O que fiz desde a última reunião? O que irei fazer até a próxima? Quais foram meus impedimentos?



# Planejamento da Sprint

- Reunião de Planejamento Parte 1:
  - Definido o que vai ser entregue na sprint.
  - PO apresenta Product Backlog ordenado e os explica;
- Reunião de Planejamento Parte 2:
  - Definido como a equipe realizará o trabalho;
  - Equipe decide como irá transformar os itens planejados em um incremento pronto ao final da sprint;



## Revisão da Sprint

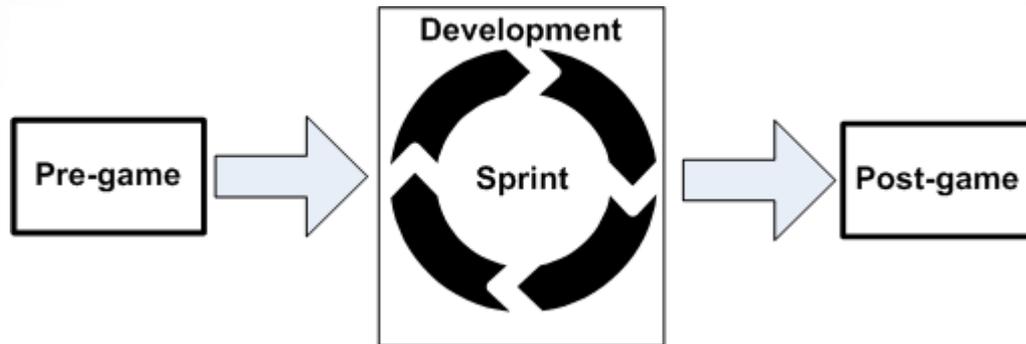
- Apresentar ao PO todos os entregáveis produzidos;
- Time se compromete a assumir responsabilidade pelo que foi e não foi produzido;
- Discutir soluções e ouvir sugestões sobre o que foi apresentado;
- PO deve validar a sprint verificando se a meta da sprint foi atingida.



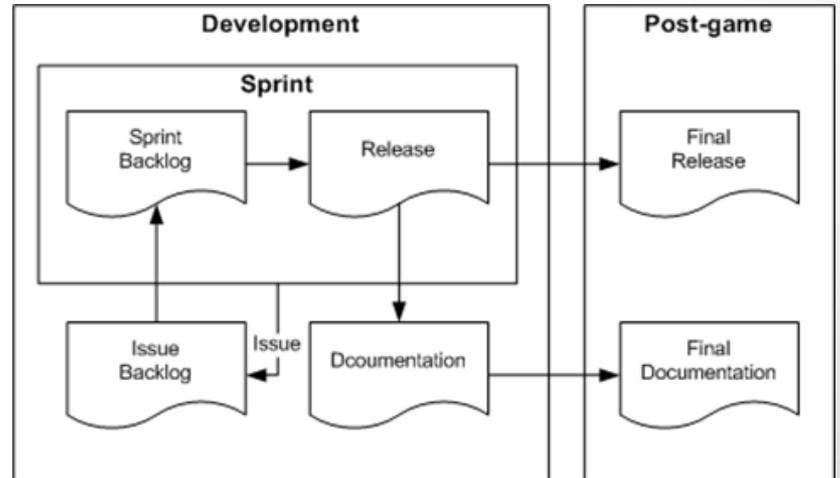
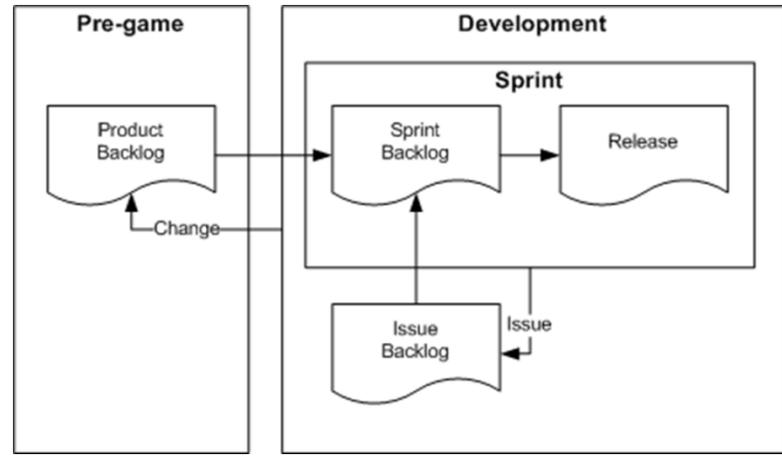
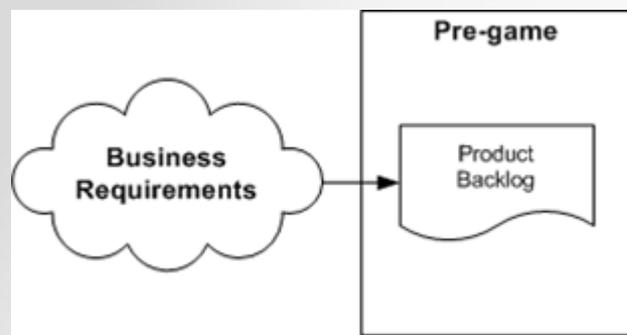
## Restrospectiva da Sprint

- Inspeccionar como a última Sprint foi em relação as pessoas, relações, processos e ferramentas;
- Identificar e ordenar os principais itens que foram bem e as potenciais melhorias;
- Criar um plano para implementar melhorias no modo que o Time Scrum faz seu trabalho;

# Ciclo Desenvolvimento



- **Pré Planejamento** : Iniciação do projeto com requisitos e planejamento de alto nível.
- **Desenvolvimento**: Múltiplos ciclos de desenvolvimento (Sprints), entregando incrementos do produto.
- **Pós Planejamento** : Disponibilização do projeto com a publicação e entrega da documentação



O ciclo de vida da metodologia Scrum se divide nas fases de pré-planejamento, desenvolvimento e pós-planejamento. O documento denominado *product backlog* é gerado na fase de desenvolvimento.

Certo

Errado

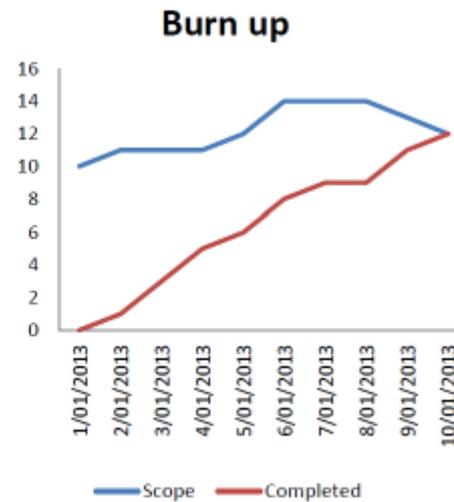
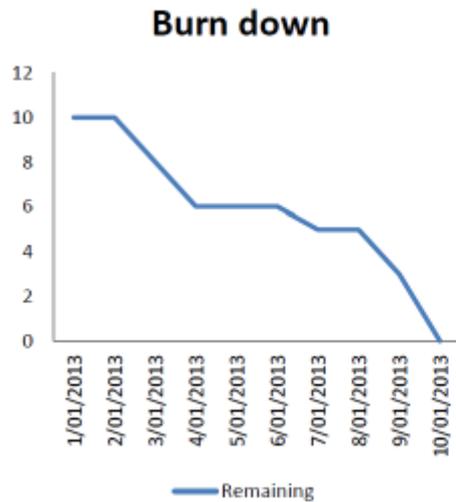
O ciclo de vida da metodologia Scrum se divide nas fases de pré-planejamento, desenvolvimento e pós-planejamento. O documento denominado *product backlog* é gerado na fase de desenvolvimento.

Certo

➔ Errado

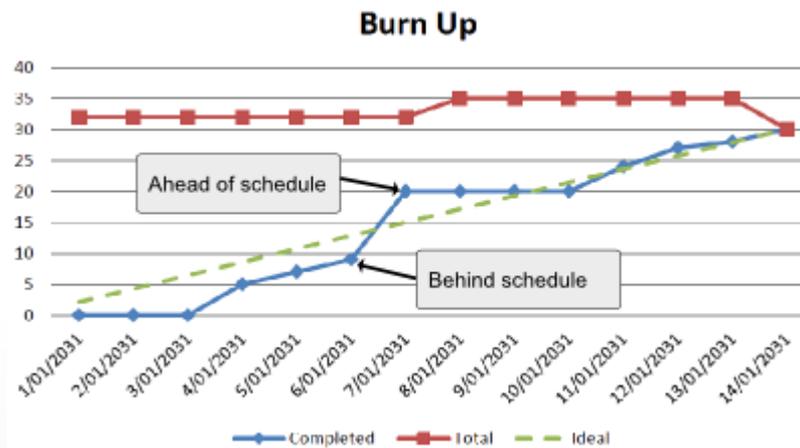
# Artefatos

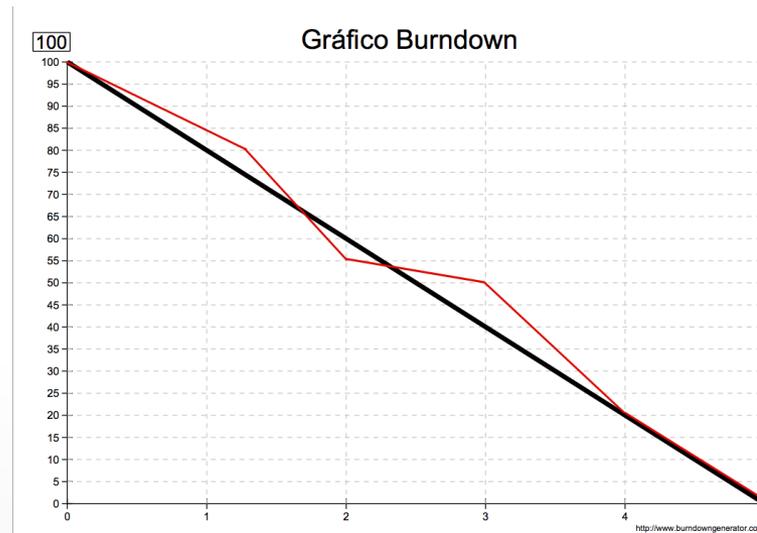
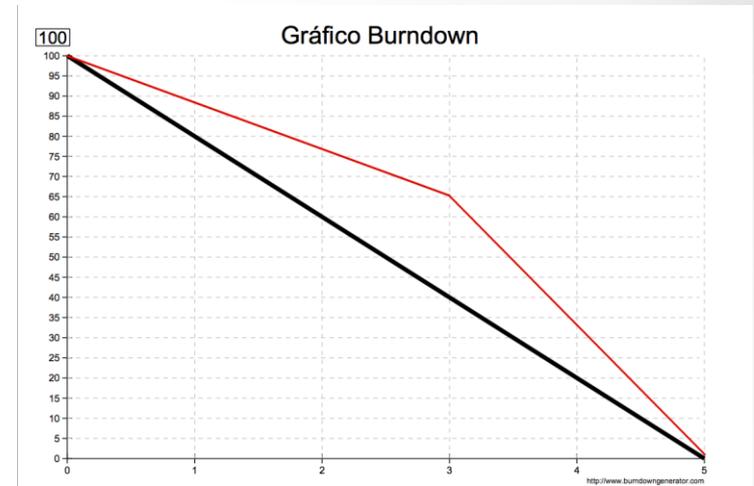
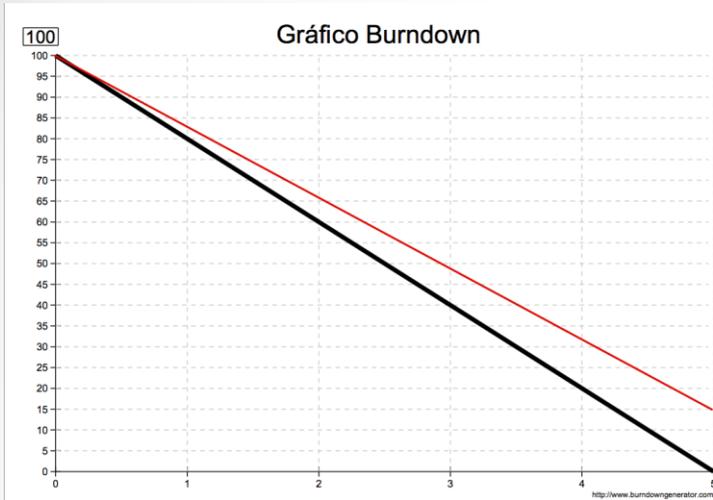
- Product Backlog
  - Lista ordenada por prioridade de todas as funcionalidades necessárias para o produto
  - Cada item tem um peso de acordo com a vontade do cliente
  - Replanejado a cada sprint
  - Responsável é o Product Owner
- Sprint Backlog
  - Lista de tarefas definidas pelo Team junto ao PO
  - Considera-se a prioridade dada pelo cliente e o tempo e esforço estimado pela equipe
- Burndown Chart
  - Gráfico para representar diariamente o progresso do trabalho em desenvolvimento;
  - Simples, porém capaz de demonstrar o andamento da Sprint

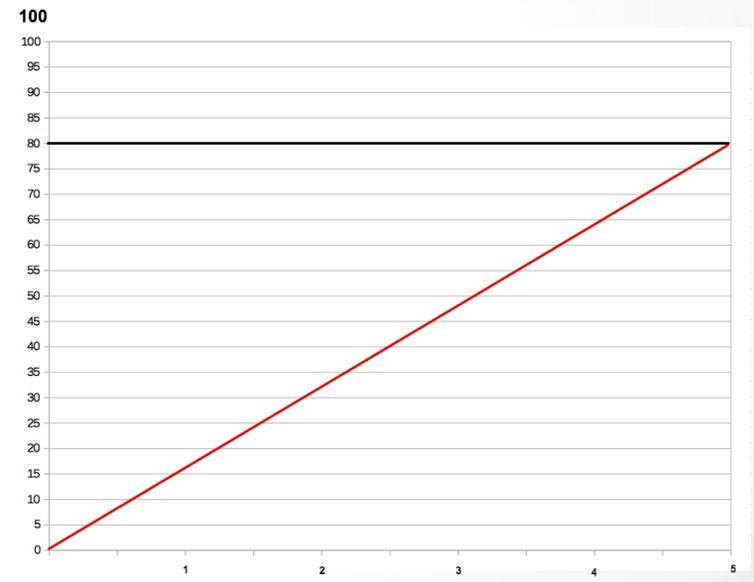
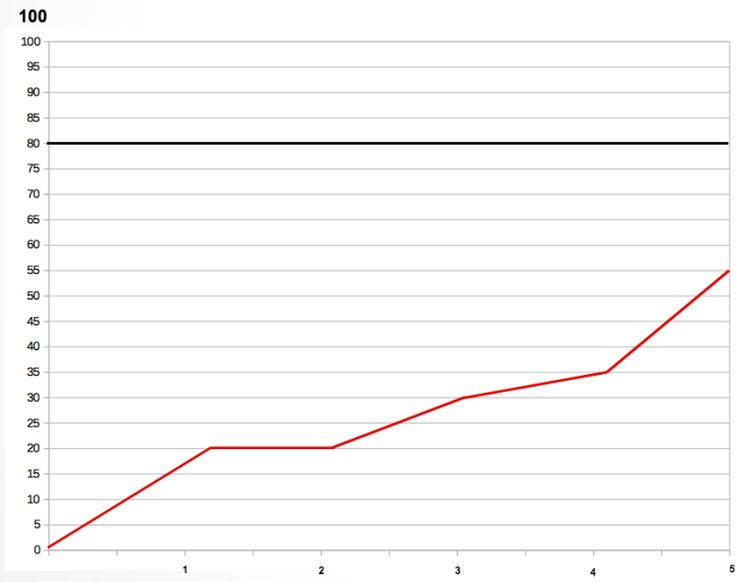
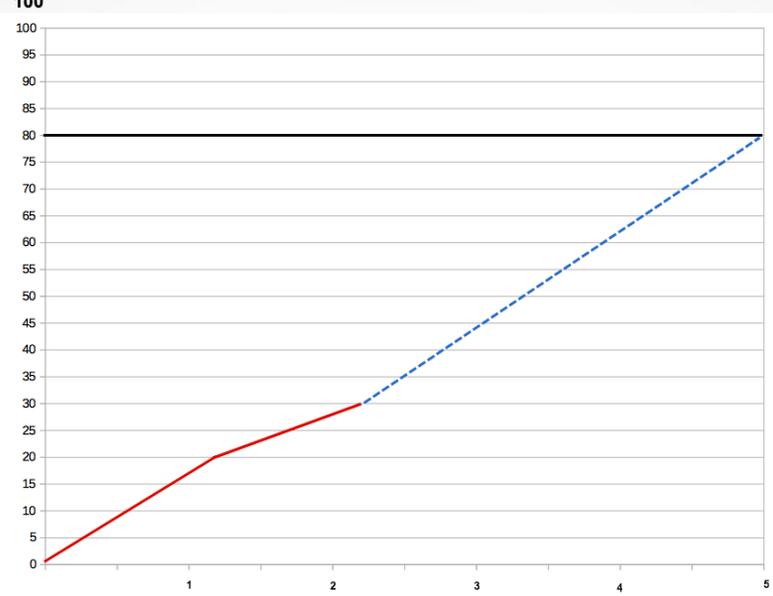


O que está sendo feito?

O que falta fazer?



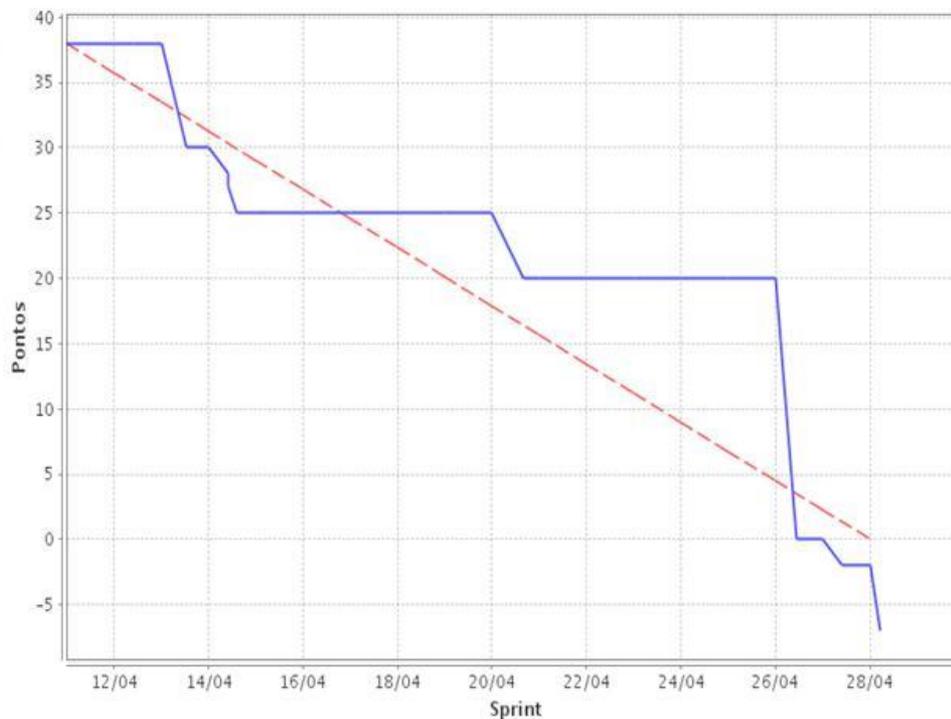




DETALHES

- Meta
- Início
- Término
- Restam
- Status
- Pontos Previstos
- Pontos Realizados

BURNDOWN



# Time Box



- Tempo máximo pré-determinado para sua realização.
- Intervalo de tempo para realização de uma atividade.
- **Objetivo:**
  - Garantir que uma quantidade de tempo adequada seja utilizada em cada evento, sem desperdiçar o tempo disponível para o projeto.

# Time Box

- Vantagens:
  - Reuniões focadas e objetivas;
  - Eliminação do desperdício de tempo;
  - Evita a ociosidade, pois todos sabem o que precisa ser feito;
  - Valorização do tempo como recurso fundamental para o sucesso do projeto;
  - Dedicção exclusiva para o objetivo;

Scrum Events	Timebox	
	2 weeks	1 month
Sprint Planning Meeting 1	2 hrs	4 hrs
Sprint Planning Meeting 2	2 hrs	4 hrs
Daily Scrum	15 min	15 min
Sprint Review	2 hrs	4 hrs
Sprint Retrospective	1,5 hrs	3 hrs

# Planning Poker



## 7 – CESPE – 2015 - TELEBRAS

O *product backlog*, lista das principais funcionalidades que o produto ou projeto tem de executar, é um dos artefatos mais importantes do *framework* ágil do *scrum*, sendo construído apenas pelo *product owner* e pelo *time* de execução. (Certo/Errado)

## 8 – CESPE – 2015 – TELEBRAS

Diferentemente da abordagem de gerência de projetos *waterfall*, que é orientada ao planejamento detalhado e de execução sequencial, o *scrum* é fundamentado em um desenvolvimento incremental e iterativo que permite maior adaptação e antecipação dos potenciais problemas. (Certo/Errado)

## 9 – CESPE – 2015 – MEC

No Scrum, os projetos são particionados em ciclos de tempo denominados *Sprints*, nas quais um conjunto de atividades deve ser executado. (Certo/Errado)

## 7 – CESPE – 2015 - TELEBRAS

O *product backlog*, lista das principais funcionalidades que o produto ou projeto tem de executar, é um dos artefatos mais importantes do *framework* ágil do *scrum*, sendo construído apenas pelo *product owner* e pelo *time* de execução. (Certo/Errado)

## 8 – CESPE – 2015 – TELEBRAS

Diferentemente da abordagem de gerência de projetos *waterfall*, que é orientada ao planejamento detalhado e de execução sequencial, o *scrum* é fundamentado em um desenvolvimento incremental e iterativo que permite maior adaptação e antecipação dos potenciais problemas. (Certo/Errado)

## 9 – CESPE – 2015 – MEC

No Scrum, os projetos são particionados em ciclos de tempo denominados *Sprints*, nas quais um conjunto de atividades deve ser executado. (Certo/Errado)

## 10 – CESPE – 2015 – MEC

*Daily Scrum*, também conhecido como *Daily Standup*, é uma reunião diária rápida para se atualizar o *Scrum Master* sobre o estado do projeto. (Certo/Errado)

## 11 – CESPE – 2013 – INPI

Conforme a metodologia SCRUM, *Sprint Planning Meeting* é uma reunião de planejamento em que o *Scrum Master* prioriza os itens do *Product Backlog* e a equipe seleciona as atividades a serem implementadas no período. (Certo/Errado)

## 12 – CESPE – 2013 – ANP

De acordo com a metodologia Scrum, a constituição ideal da equipe de desenvolvimento para que o trabalho se mantenha ágil deve ser de menos de três pessoas. (Certo/Errado)

## 10 – CESPE – 2015 – MEC

Daily *Scrum*, também conhecido como *Daily Standup*, é uma reunião diária rápida para se atualizar o *Scrum Master* sobre o estado do projeto. (Certo/Errado)

## 11 – CESPE – 2013 – INPI

Conforme a metodologia SCRUM, *Sprint Planning Meeting* é uma reunião de planejamento em que o *Scrum Master* prioriza os itens do *Product Backlog* e a equipe seleciona as atividades a serem implementadas no período. (Certo/Errado)

## 12 – CESPE – 2013 – ANP

De acordo com a metodologia Scrum, a constituição ideal da equipe de desenvolvimento para que o trabalho se mantenha ágil deve ser de menos de três pessoas. (Certo/Errado)

# ASD (Adaptative Software Development)

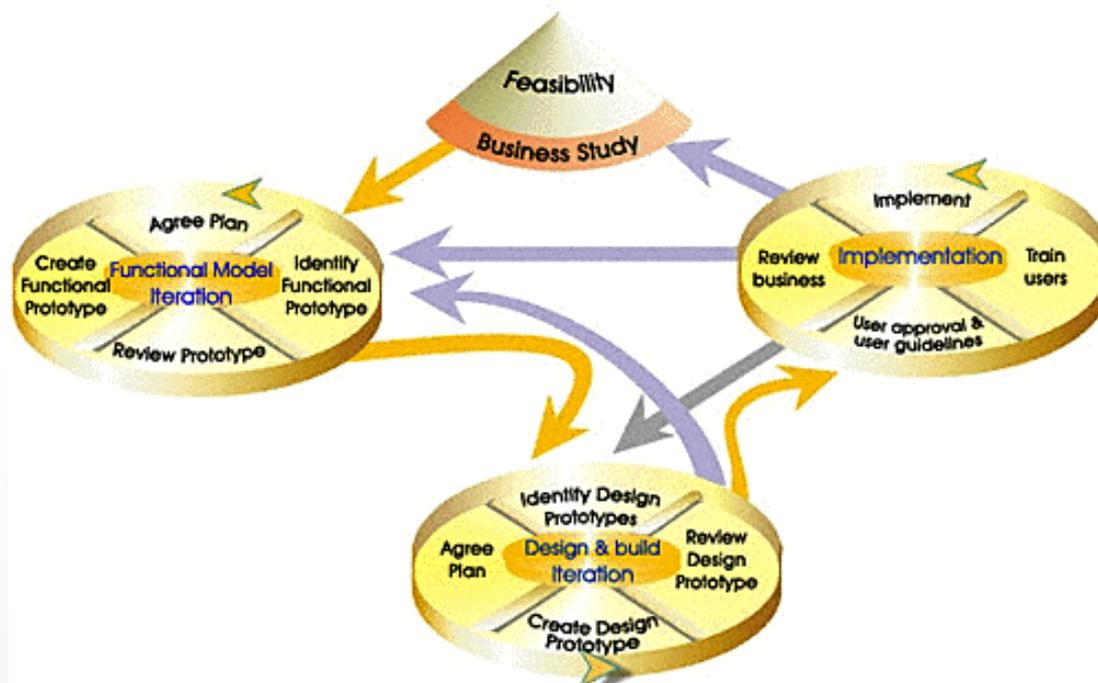
- Técnica para construção de software complexos;
- Bases se concentram na colaboração humana e na auto-organização das equipes;
- Ciclo de vida com 3 fases: especulação, colaboração e aprendizagem;
- Enfatiza o aprendizado e o progresso para um ciclo completo;
- Adaptação contínua do processo de acordo com o trabalho em questão;

# DSDM – Dynamic Systems Development Method

- Método de Desenvolvimento Dinâmico de Sistemas;
- Oferece uma metodologia para se trabalhar com restrições de prazo;
- Utiliza prototipagem incremental em um ambiente controlado;
- Baseia-se no princípio de Pareto (80/20);
- Cada iteração segue a regra dos 80%;
- Aspectos técnicos são pouco abordados;

# DSDM – Dynamic Systems Development Method

- As quatro fases são antecedidas pelo pré-projeto e sucedidas pelo pós-projeto

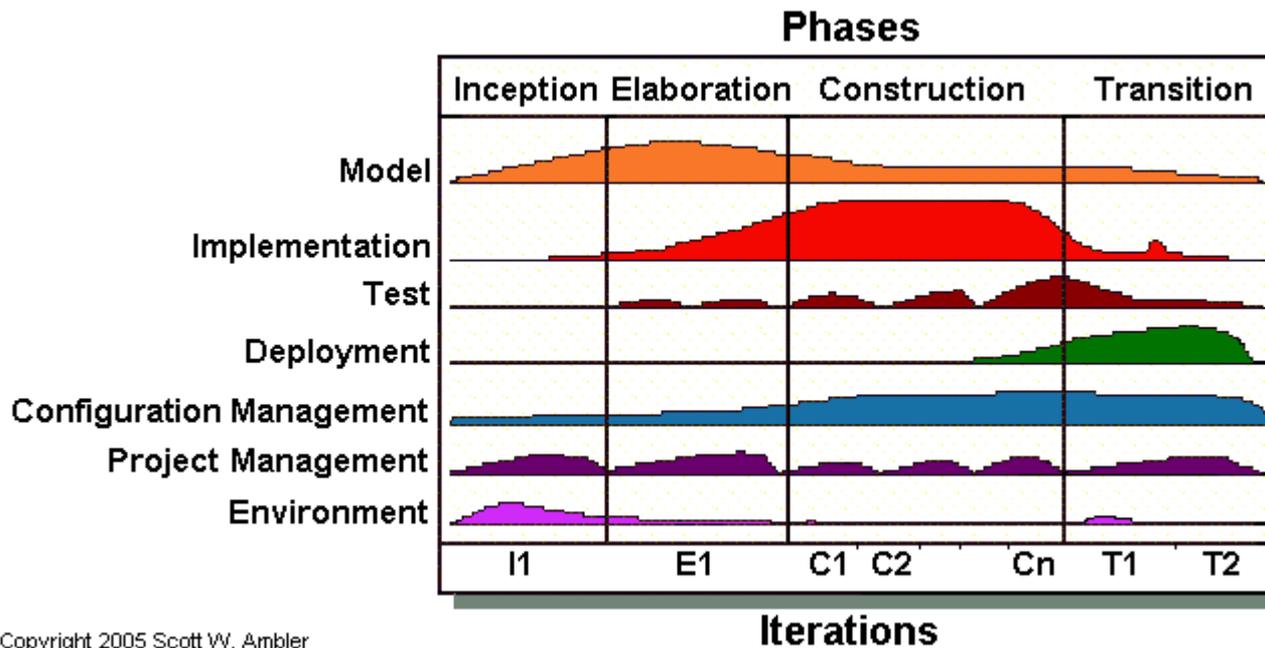


# DSDM – Dynamic Systems Development Method

- **Pré-Projeto**
  - Tem como objetivo definir se o projeto será ou não implementado sob aspectos gerenciais
- **Pós-Projeto**
  - Manter o sistema através de tarefas de alteração no sistema
- 1. Estudo de Viabilidade
  - Avalia as necessidades de informação para o escopo
- 2. Modelo Funcional
  - Obtém requisitos funcionais e não funcionais de forma iterativa e incremental
- 3. Design e Construção
  - Detalha os requisitos e desenvolve o sistema
- 4. Implementação
  - Transição do sistema para o ambiente operacional

# AUP – Agile Unified Process

- Abordagem simplificada para o desenvolvimento de software com base no RUP



# AUP – Agile Unified Process (Fases)

- 1. Iniciação** (Objetivos do Ciclo de Vida)
  - Identificar o escopo inicial e a possível arquitetura do sistema
- 2. Elaboração** (Ciclo de Vida da Arquitetura)
  - Provar a arquitetura do sistema e criar um protótipo da mesma
- 3. Construção** (Capacidade Operacional Inicial)
  - Escrever os códigos do sistema de forma incremental
- 4. Transição** (Lançamento do Produto)
  - Validar e implantar o sistema no ambiente de produção

# AUP – Agile Unified Process (Disciplinas)

## 1. Modelo

- Identificar uma solução viável para resolver o problema abordado

## 2. Implementação

- Transformar os modelos em códigos executáveis, incluindo testes unitários

## 3. Teste

- Encontrar defeitos durante a validação do sistema

## 4. Implantação

- Planejar a entrega do sistema e executar o plano para disponibilização final

## 5. Gerenciamento de Configuração

- Controlar as versões do projeto ao longo do tempo

## 6. Gerenciamento de Projetos

- Dirigir atividades do projeto, tais como gerenciamento de riscos, pessoas, escopo, entre outras

## 7. Ambiente

- Apoiar as demais e garantir que as ferramentas estejam disponíveis para a equipe quando necessário

# AUP – Agile Unified Process (Marcos)

## 1. Iniciação

- Identificar o escopo inicial e a possível arquitetura do sistema

## 2. Elaboração

- Provar a arquitetura do sistema e criar um protótipo da mesma

## 3. Construção

- Escrever os códigos do sistema de forma incremental

## 4. Transição

- Validar e implantar o sistema no ambiente de produção

# FDD – Feature Driven Development

- Desenvolvimento Dirigido por Funcionalidade
- Focado na entrega regular de funcionalidades para o cliente;
- Equipes podem variar de 10 a 250 programadores;
- Menos detalhado que outras metodologias XP;
- Funcionalidade = “uma função valorizada pelo cliente passível de ser implementada em duas semana ou menos”;
- Relatórios de acompanhamento;

# FDD – Feature Driven Development

## - Práticas -

1. Equipes Modelam com Base no Negócio. Toda a equipe trabalha em conjunto no entendimento do negócio. Não existe distinção entre equipe de negócio e desenvolvimento.
2. Desenvolvimento por Funcionalidades. Implementação orientada por funcionalidades, com foco no desenvolvimento de entregas em intervalos curtos.
3. Propriedade Individual. Código é escrito e mantido apenas pelo seu “dono”, com o objetivo de ter maior rapidez na implementação.
4. Equipes Trabalham por Funcionalidade. Cada funcionalidade tem sua equipe especialista.
5. Inspeções de Qualidade. Inspeções como forma de verificar a qualidade do código.
6. Integrações e Builds Frequentes. Funcionalidades finalizadas devem ser integradas constantemente.
7. Gerenciamento de Configuração. Foco em manter versões de todas as funcionalidades criadas e garantir a integridade dos itens de configuração.
8. Transparência do Progresso. Garantir que todos conheçam o progresso e os resultados esperados.

# FDD – Feature Driven Development

## - Processos -

- **Desenvolver um Modelo Abrangente**
  - Desenvolvimento de requisitos, análise OO, modelagem e outras técnicas com objetivo de obter um modelo de objetos de alto nível;
- **Construir uma Lista de Funcionalidades**
  - Decomposição funcional do modelo de domínio tendo como resultado uma hierarquia de funcionalidades representando o produto a ser desenvolvido;
- **Planejar por Funcionalidades**
  - Estimar a complexidade e dependência das funcionalidades, levando em consideração a prioridade e valor para o negócio
- **Detalhar por Funcionalidade**
  - Detalhamento de requisitos e outros artefatos para construir a funcionalidade
- **Construir por Funcionalidade**
  - Cada esqueleto de código é preenchido, testado e inspecionado, resultando em um incremento do produto integrado ao repositório.

# FDD – Feature Driven Development

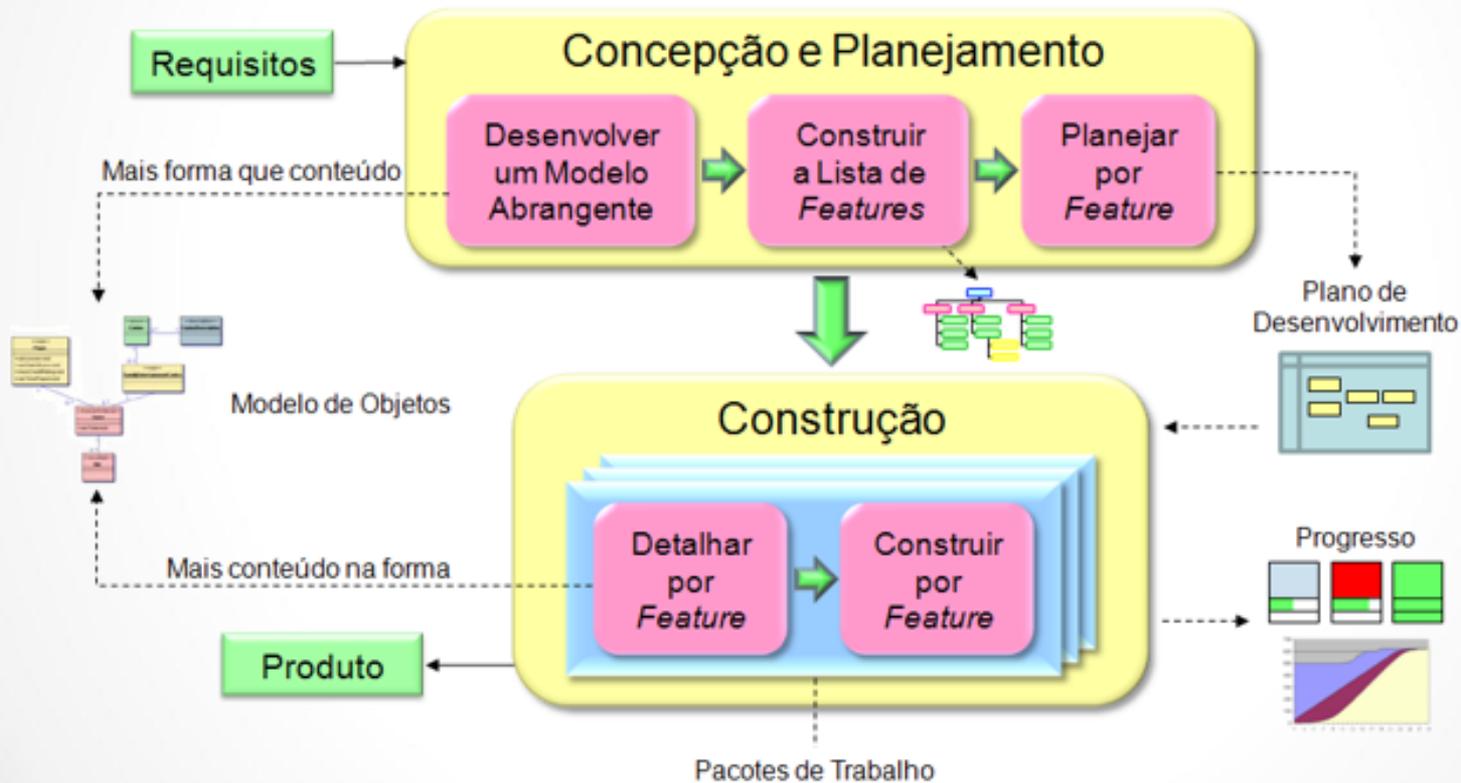
## - Marcos de Referência -

- Travessia do Projeto,
  - Projeto,
  - Inspeção do Projeto,
  - Código,
  - Inspeção do Código,
  - Promoção para Construção.
- As três primeiras são finalizadas durante o “Planejar por Funcionalidade” e as últimas três são completadas durante “Construir por Funcionalidade”

Table 1: Milestones

Domain Walkthrough	Design	Design Inspection	Code	Code Inspection	Promote To Build
1%	40%	3%	45%	10%	1%

# FDD – Feature Driven Development



Há um considerável debate sobre os benefícios e a aplicabilidade do desenvolvimento ágil de software em contraposição aos processos mais convencionais de engenharia de software. Relacione o modelo ágil de software com a sua respectiva característica.

### **Modelo**

I - DAS II - DSDM III - FDD IV - XP

### **Característica**

(P)

Define um ciclo de vida que incorpora três fases: especulação, colaboração e aprendizado. Durante a fase de aprendizado, à medida que os membros de uma equipe começam a desenvolver os componentes que fazem parte de um ciclo adaptativo, a ênfase está tanto no aprendizado quanto no progresso em direção a um ciclo completo.

(Q)

O conceito característica é uma função valorizada pelo cliente, que pode ser implementada em duas semanas ou menos. Este modelo define seis marcos de referência durante o projeto e implementação de uma característica: travessia do projeto, projeto, inspeção de projeto, código, inspeção de código, promoção para construção.

(R)

Fornecer um arcabouço para construir e manter sistemas que satisfazem às restrições de prazo apertadas por meio do uso de prototipagem incremental em ambiente controlado de projeto. Essa abordagem sugere uma filosofia que é emprestada de uma versão modificada do princípio de Pareto.

A relação correta é:

- a) I - P, II - Q, III - R.
- b) I - P, II - R, III - Q.
- c) I - Q, III - R, IV - P.
- d) II - P, III - R, IV - Q.
- e) II - Q, III - P, IV - R.

Há um considerável debate sobre os benefícios e a aplicabilidade do desenvolvimento ágil de software em contraposição aos processos mais convencionais de engenharia de software. Relacione o modelo ágil de software com a sua respectiva característica.

### **Modelo**

I - DAS II - DSDM III - FDD IV - XP

### **Característica**

(P)

Define um ciclo de vida que incorpora três fases: especulação, colaboração e aprendizado. Durante a fase de aprendizado, à medida que os membros de uma equipe começam a desenvolver os componentes que fazem parte de um ciclo adaptativo, a ênfase está tanto no aprendizado quanto no progresso em direção a um ciclo completo.

(Q)

O conceito característica é uma função valorizada pelo cliente, que pode ser implementada em duas semanas ou menos. Este modelo define seis marcos de referência durante o projeto e implementação de uma característica: travessia do projeto, projeto, inspeção de projeto, código, inspeção de código, promoção para construção.

(R)

Fornece um arcabouço para construir e manter sistemas que satisfazem às restrições de prazo apertadas por meio do uso de prototipagem incremental em ambiente controlado de projeto. Essa abordagem sugere uma filosofia que é emprestada de uma versão modificada do princípio de Pareto.

A relação correta é:

- a) I - P, II - Q, III - R.
- b) I - P, II - R, III - Q.
- c) I - Q, III - R, IV - P.
- d) II - P, III - R, IV - Q.
- e) II - Q, III - P, IV - R.

O conceito de sprint aplica-se ao modelo ágil do processo de engenharia de software denominado

- a) XP.
- b) DAS.
- c) DSDM.
- d) Scrum.
- e) Crystal.

O conceito de sprint aplica-se ao modelo ágil do processo de engenharia de software denominado

- a) XP.
- b) DAS.
- c) DSDM.
- d) Scrum.
- e) Crystal.

FDD (*Feature Driven Development*) é uma metodologia muito objetiva, possuindo apenas duas fases:

- a) Concepção & Planejamento e Construção.
- b) Decomposição Funcional e Construção.
- c) Análise dos Requisitos e Desenvolvimento.
- d) Planejamento Incremental e Desenvolvimento por Funcionalidade.
- e) Planejamento por Funcionalidade e Construção por Funcionalidade.

FDD (*Feature Driven Development*) é uma metodologia muito objetiva, possuindo apenas duas fases:

- ➔ a) Concepção & Planejamento e Construção.
- b) Decomposição Funcional e Construção.
- c) Análise dos Requisitos e Desenvolvimento.
- d) Planejamento Incremental e Desenvolvimento por Funcionalidade.
- e) Planejamento por Funcionalidade e Construção por Funcionalidade.

A Feature Driven Development (FDD) é uma metodologia ágil de desenvolvimento de software, sobre a qual é correto afirmar:

- a) Não pode ser combinada a outras técnicas para a produção de sistemas.
- b) Possui cinco processos: Desenvolver um Modelo Abrangente, Construir a Lista de Funcionalidades, Planejar por Funcionalidade, Detalhar por Funcionalidade e Implementar por Funcionalidade.
- c) Divide os papéis em dois grupos: papéis chave e papéis de apoio. Dentro de cada categoria, os papéis são atribuídos a um único participante que assume a responsabilidade pelo papel.
- d) Mantém seu foco apenas na fase de modelagem.
- e) Mantém seu foco apenas na fase de implementação.

A Feature Driven Development (FDD) é uma metodologia ágil de desenvolvimento de software, sobre a qual é correto afirmar:

a) Não pode ser combinada a outras técnicas para a produção de sistemas.

→ b) Possui cinco processos: Desenvolver um Modelo Abrangente, Construir a Lista de Funcionalidades, Planejar por Funcionalidade, Detalhar por Funcionalidade e Implementar por Funcionalidade.

c) Divide os papéis em dois grupos: papéis chave e papéis de apoio. Dentro de cada categoria, os papéis são atribuídos a um único participante que assume a responsabilidade pelo papel.

d) Mantém seu foco apenas na fase de modelagem.

e) Mantém seu foco apenas na fase de implementação.

Os modelos ágeis de desenvolvimento de software têm menos ênfase nas definições de atividades e mais ênfase na pragmática e nos fatores humanos do desenvolvimento. Um destes modelos enfatiza o uso de orientação a objetos e possui apenas duas grandes fases: 1 - Concepção e Planejamento e 2 - Construção. A fase de Concepção e Planejamento possui três disciplinas (chamadas de processos): Desenvolver Modelo Abrangente, Construir Lista de Funcionalidades e Planejar por funcionalidade. Já a fase de Construção incorpora duas disciplinas (processos): Detalhar por Funcionalidade e Construir por Funcionalidade.

O texto acima apresenta a metodologia ágil conhecida como

- a) XP.
- b) Scrum
- c) Crystal Clear.
- d) ASD
- e) FDD

Os modelos ágeis de desenvolvimento de software têm menos ênfase nas definições de atividades e mais ênfase na pragmática e nos fatores humanos do desenvolvimento. Um destes modelos enfatiza o uso de orientação a objetos e possui apenas duas grandes fases: 1 - Concepção e Planejamento e 2 - Construção. A fase de Concepção e Planejamento possui três disciplinas (chamadas de processos): Desenvolver Modelo Abrangente, Construir Lista de Funcionalidades e Planejar por funcionalidade. Já a fase de Construção incorpora duas disciplinas (processos): Detalhar por Funcionalidade e Construir por Funcionalidade.

O texto acima apresenta a metodologia ágil conhecida como

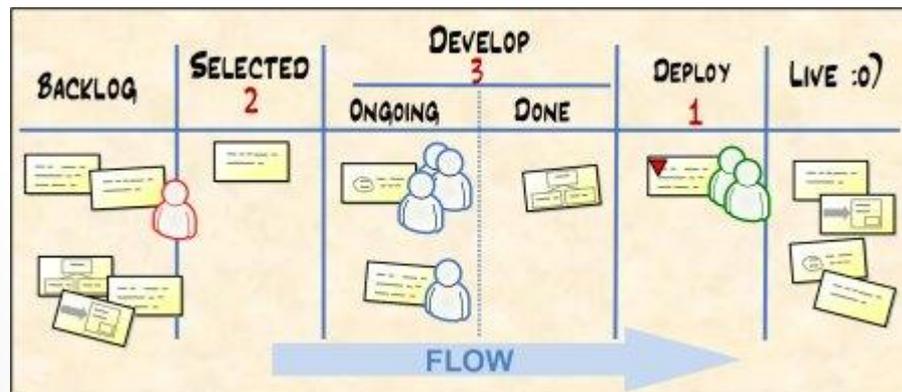
- a) XP.
- b) Scrum
- c) Crystal Clear.
- d) ASD
- e) FDD

# LSD – Lean Software Development

- Desenvolvimento de Software Enxuto;
- Adaptou os princípios de fabricação enxuta para a engenharia de software;
- Princípios:
  - Eliminar Desperdício,
  - Incorporar Qualidade,
  - Criar Conhecimento,
  - Adiar Compromissos,
  - Entregar Rápido,
  - Respeitar as Pessoas
  - Otimizar o Todo
- Sistema Toyota de Produção
  - Aumentar a eficiência da produção pela eliminação de desperdícios
- Just in Time
  - Sistema de administração de produção;
  - Nada deve ser produzido, transportado ou comprado antes da hora exata;
  - Reduz estoques e os custos decorrentes;
  - Kanban pode ser usado como Ferramenta

# Kanban

- Registro, ou placa visível;
- Time trabalha apenas em atividades solicitadas pelo cliente;
- Objetiva eliminar desperdício e aumentar o desempenho na resposta a itens específicos;
- Não define um processo de Desenvolvimento (se adequa ao existente)
- Formato:
  - Time Exclusivo
  - Time Compartilhado



O sistema denominado *kanban* tem por objetivo controlar e balancear a produção, com eliminação dos desperdícios, e acionar um sistema de reposição de estoque pela indicação dos seguintes fatores: o que, quando e quanto fornecer e produzir.

Certo

Errado

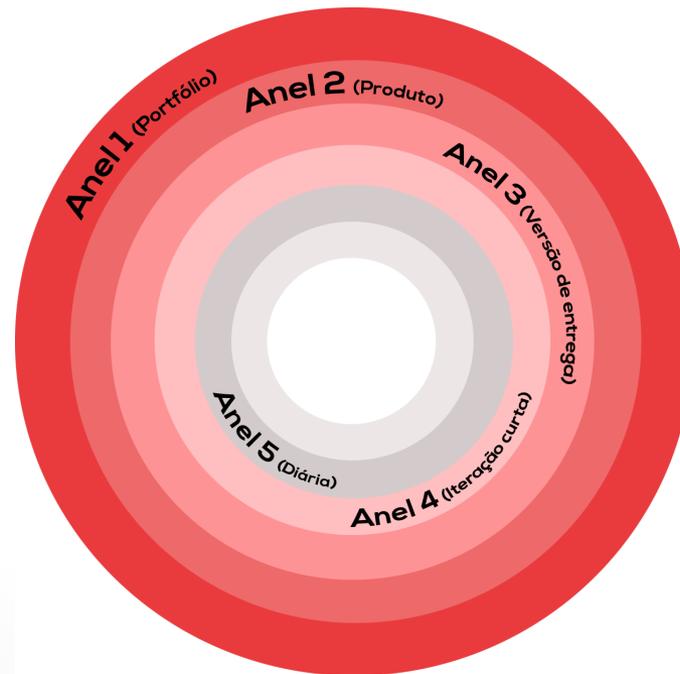
O sistema denominado *kanban* tem por objetivo controlar e balancear a produção, com eliminação dos desperdícios, e acionar um sistema de reposição de estoque pela indicação dos seguintes fatores: o que, quando e quanto fornecer e produzir.

→ Certo

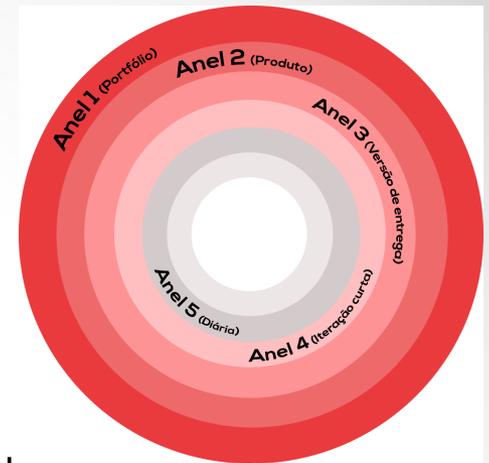
Errado

# Planning Onion

- Conhecido também como planejamento Ágil;
- Os anéis representam os níveis de planejamento que devemos realizar em nossos projetos;
- Segue a ordem de fora para dentro, sendo o maior anel superficial, e quanto mais interno mais detalhado e específico;



# Planning Onion



- **Anel 1 – Planejamento do Portfólio**
  - Representa o portfólio de projetos;
  - Quais projetos deverão ser atendidos;
  - Informações macros, tais como orçamento, prazo inicial e final, e patrocinador.
- **Anel 2 – Planejamento do Produto ou Projeto**
  - Produto propriamente dito;
  - Informações superficiais do projeto, tais como necessidades básicas, expectativas e especificações mínimas;
- **Anel 3 – Planejamento da Versão da Entrega**
  - Detalhes mais específicos referentes à equipe, ao tempo e requisitos mínimos que o produto deverá ter;
  - Iterações curtas para desenvolver o pedaço do produto;
- **Anel 4 – Planejamento da Iteração**
  - Time constrói uma parte menor do produto;
- **Anel 5 – Planejamento do Dia**
  - Composto pelos dias de trabalho;
  - Ocorre para que o time inspecione e adapte o trabalho que está sendo executado no dia-a-dia;

# Crystal

- Se refere a uma família de métodos ágeis;
- Conjunto de processos ágeis que provaram ser efetivos para diferentes tipos de projetos;
- Intenção é permitir que equipes ágeis selecionem o membro da família Crystal mais apropriado;
- De acordo com 3 parâmetros:
  - 1. A carga da comunicação**
  - 2. A criticidade do Sistema**
  - 3. A prioridade do Projeto**

# Crystal

- Carga de Comunicação:
  - Representada pelo número de pessoas na equipe. Quanto menos integrantes, menor a carga de comunicação;
- Criticidade do Sistema:
  - Riscos que o sistema pode causar. Grau de dano que pode ser acarretado pelo mau funcionamento do sistema;
- Prioridade do Projeto:
  - Importância que ele tem para a organização e suas pessoas. O quão importante é a necessidade deste ser executado o mais breve possível;

# Crystal

	Clear	Yellow	Orange	Red	Maroon
Life (L)	L6	L20	L40	L80	L200
Essential Money (E)	E6	E20	E40	E80	E200
Discretionary Money (D)	D6	D20	D40	D80	D200
Comfort (C)	C6	C20	C40	C80	C200
	1-6	7-20	21-40	41-80	81-200

# Crystal - Princípios

- **Entrega Frequente**
  - Garantir a entrega de software funcionando em um curto período
- **Melhorias Reflexivas**
  - Equipe se reúne para discutir como reverter falhas em sucesso.
- **Comunicação Osmótica**
  - Informação que surge naturalmente a partir da comunicação entre a equipe
- **Segurança Pessoal**
  - Dizer sempre quando algo está incomodando
- **Foco**
  - Trabalhar naquilo que foi demandado
- **Fácil Acesso aos Especialistas**
  - Equipe precisa ter acesso a especialistas sempre que for necessário
- **Excelência Técnica**
  - Testes automatizados, gerenciamento de configuração e integração contínua

As práticas se baseiam em técnicas ágeis, tais como, Test Driven Development (TDD), Agile Model Driven Development (AMDD) e Database Refactoring, concentrando as atividades de análise, desenho e requisitos unicamente na disciplina Modelagem. Trata-se de

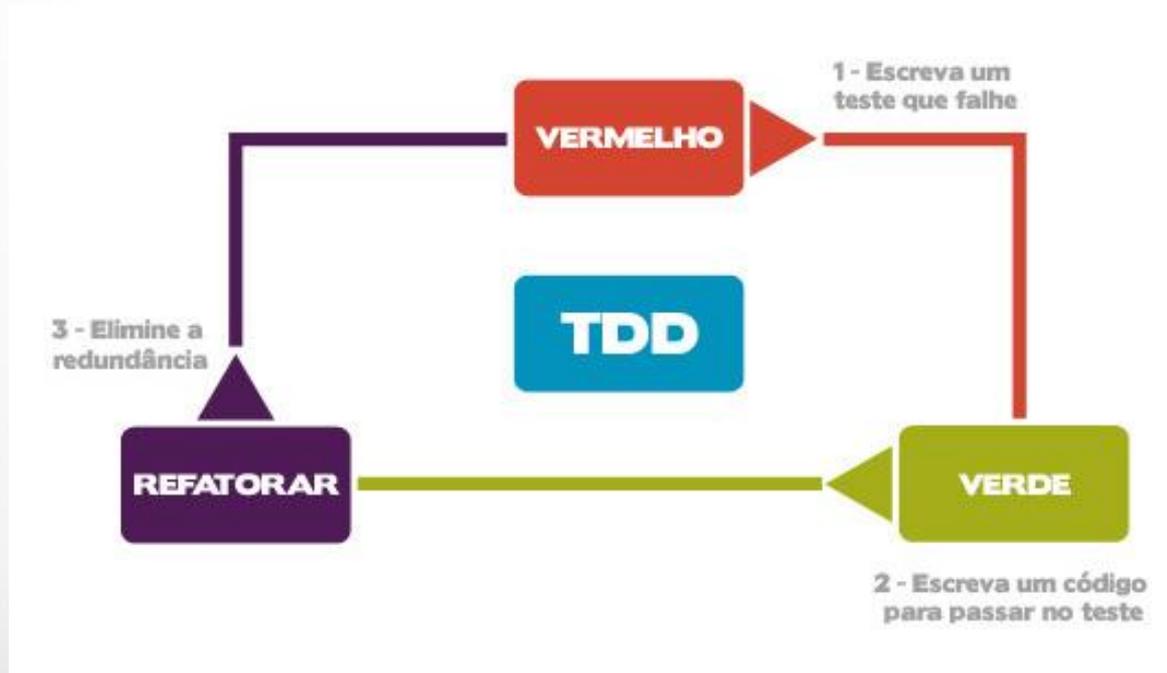
- a) AUP.
- b) SCRUM.
- c) XP.
- d) AUP e SCRUM.
- e) XP e SCRUM.

As práticas se baseiam em técnicas ágeis, tais como, Test Driven Development (TDD), Agile Model Driven Development (AMDD) e Database Refactoring, concentrando as atividades de análise, desenho e requisitos unicamente na disciplina Modelagem. Trata-se de

- ➔ a) AUP.
- b) SCRUM.
- c) XP.
- d) AUP e SCRUM.
- e) XP e SCRUM.

# TDD

- Utilizado principalmente para garantir a qualidade dos produtos desenvolvidos e minimizar o risco de defeitos no produto final;
- Mudar a forma de pensar, escrevendo primeiro o teste e depois o código;



No TDD, o refatoramento do código deve ser realizado antes de se escrever a aplicação que deve ser testada.

Certo

Errado

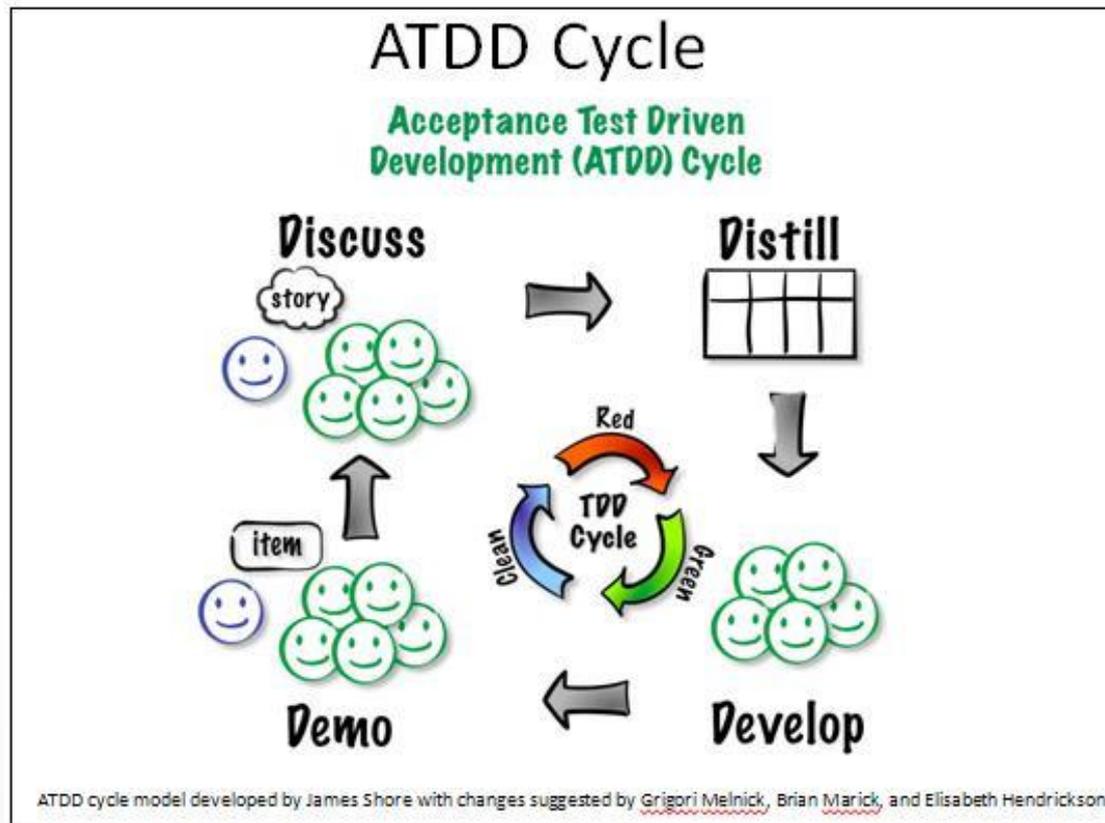
No TDD, o refatoramento do código deve ser realizado antes de se escrever a aplicação que deve ser testada.

Certo

→ Errado

# ATDD

- Foca nos testes sob o ponto de vista do cliente



# ATDD

- Discute
  - Todas as histórias definidas como requisitos das funcionalidades a serem construídas precisam ser levadas para o cliente com o objetivo de definir os critérios de aceitação
- Detalha
  - Documenta os teste de aceitação a fim de padronizar e formalizar as necessidades definidas e identificadas como requisitos
- Desenvolve
  - Escrever os testes de aceitação com o foco no atendimento de requisitos do negócio do cliente
- Demonstra
  - Demonstra o produto pronto de acordo com os testes de aceitação escritos

# BDD – Behavior Driven Development

- Criado como uma resposta ao TDD
- Software é desenvolvido de “Fora para Dentro”.
- Objetiva integrar regras de negócios com linguagem de programação, focando no comportamento do software.
- Garantir que a aplicação está sendo desenvolvida de acordo com o teste de aceitação escrito.
- Linguagem Ubiqua – Todos devem falar a mesma língua

# BDD – Behavior Driven Development

## - Descrição -

- *Funcionalidade : [Nome]*
- *Para [ Valor ao Negócio ]*  
*Eu, como [ Papel ]*  
*Desejo poder realizar [ Funcionalidade ]*
  
- *Cenário : [ Nome ]*
- *Dado que [ Estado inicial do sistema ]*  
*Quando [ Ação a ser realizada no sistema ]*  
*Então [ Coisas que o sistema deve fazer após a ação do*  
*Quando ]*

# BDD – Behavior Driven Development

## - Exemplo -

#language: pt-br

Funcionalidade: Vender doces

*Para quando uma doce for vendido*

*Eu, como vendedor*

*Desejo decrementar um item no estoque*

Cenário: Baixa 1 bala do estoque

Dado que cliente pede 1 bala

E tenho 10 balas em estoque

Quando ele compra realiza a compra

Então eu fico com 9 balas em estoque

# Histórias INVEST

- Sugere uma estrutura normalizada e eficiente para as histórias, facilitando seu entendimento
- I – Independente
- N – Negociável
- V – Valiosa
- E – Estimável
- S – Scalable ( Dimensionada corretamente)
- T - Testável

# Tarefas SMART

- Descre uma tarefa de forma que ela seja específica, mensurável, realizável, relevante e que tenha uma duração fixa.
- S – Specific
- M – Mensurável
- A – Achievable
- R – Relevante
- T – Time-Boxed

## Estimativas por Afinidade

- Ação de determinar o tamanho ou esforço de histórias com classificações similares

<b>TAMANHOS E MEDIDAS</b>					
<b>Tam.</b>	Ombro	Tórax	Comp.	Manga	O+M
<b>PP</b>	40	100	68	23	63
<b>P</b>	42	104	70	24	66
<b>M</b>	44	108	72	25	69
<b>G</b>	46	112	74	26	72
<b>GG</b>	48	116	76	27	75

# Dias Ideais

- Um dia ideal é aquele em que uma quantidade de trabalho é realizada sem qualquer interrupção ou distração.
- Considerado um dia em que nada dá errado e até a inspiração do profissional está alta.
- Algumas equipes promovem um dia ideal onde toda a equipe vira servidora de apenas um desenvolvedor, atendendo tudo o que for necessário para atender o trabalho.

# Calendário Niko-Niko

- Niko -> Sorriso
- Niko-Niko -> “emoticon”
- A idéia é colher feedbacks diários a respeito do clima de trabalho e de possíveis causas para o aumento de defeitos, falta de motivação ou baixo desempenho.

SEPTEMBER 2011							
	1	2	3	4	5	6	7
BERND							
MARIKO							
ALEX							
MEIKE							
STEFAN							

# Gabarito

1 – ERRADO	6 – ERRADO	11 – ERRADO	16 – A
2 – ERRADO	7 – ERRADO	12 – ERRADO	17 – E
3 – CERTO	8 – CERTO	13 – B	18 – CERTO
4 – CERTO	9 – CERTO	14 – D	19 – A
5 – ERRADO	10 – ERRADO	15 – A	20 - ERRADO

# Questões Comentadas

Assinale a opção correta relativamente ao Scrum.

- a) Fixadas as datas de início e fim de uma sprint, a equipe de desenvolvimento busca cumprir o sprint backlog, estágio em que o product owner não pode negociar o escopo do backlog da sprint mesmo que tal prática reduza o ritmo do desenvolvimento.
- b) A qualquer momento após o início do projeto, geralmente depois de analisar o retorno sobre o investimento e avaliar se um conjunto de funcionalidades já pode ser utilizado por clientes e usuários, o product owner pode decidir o tempo de entrega de uma versão.
- c) Burndown e burnside são duas das práticas de estimativa utilizadas no Scrum.
- d) A definição de requisitos e a realização do trabalho no projeto ocorrem em interações, ao final das quais o produto estará pronto para que o cliente homologue as implementações por ele fixadas.
- e) Ordenados os requisitos do product backlog pelo time de desenvolvimento, o product owner avaliará a qualidade dos produtos entregues para certificar-se de que os desenvolvedores realizaram adequadamente as avaliações de mercado e as necessidades dos clientes com relação ao produto.

Assinale a opção correta relativamente ao Scrum.

- a) Fixadas as datas de início e fim de uma sprint, a equipe de desenvolvimento busca cumprir o sprint backlog, estágio em que o product owner não pode negociar o escopo do backlog da sprint mesmo que tal prática reduza o ritmo do desenvolvimento.
- ➔ b) A qualquer momento após o início do projeto, geralmente depois de analisar o retorno sobre o investimento e avaliar se um conjunto de funcionalidades já pode ser utilizado por clientes e usuários, o product owner pode decidir o tempo de entrega de uma versão.
- c) Burndown e burnside são duas das práticas de estimativa utilizadas no Scrum.
- d) A definição de requisitos e a realização do trabalho no projeto ocorrem em interações, ao final das quais o produto estará pronto para que o cliente homologue as implementações por ele fixadas.
- e) Ordenados os requisitos do product backlog pelo time de desenvolvimento, o product owner avaliará a qualidade dos produtos entregues para certificar-se de que os desenvolvedores realizaram adequadamente as avaliações de mercado e as necessidades dos clientes com relação ao produto.

No que se refere a métodos ágeis de desenvolvimento de sistemas, assinale a opção correta.

- a) A aplicação de método ágil para desenvolvimento de grandes sistemas pode enfrentar dificuldades que o tornem inviável.
- b) O documento de requisitos, apesar de abordar um conjunto pequeno de funcionalidades, deve especificar toda a necessidade do usuário.
- c) O sistema é construído em pequenos blocos, que irão compor uma versão a ser entregue aos usuários.
- d) A documentação de projeto deve ser feita pelo próprio desenvolvedor, seguindo padrões simplificados.
- e) Para atingir os objetivos de agilidade exigidos, os desenvolvedores devem seguir processos simplificados para a construção do software.

No que se refere a métodos ágeis de desenvolvimento de sistemas, assinale a opção correta.

- a) A aplicação de método ágil para desenvolvimento de grandes sistemas pode enfrentar dificuldades que o tornem inviável.
- b) O documento de requisitos, apesar de abordar um conjunto pequeno de funcionalidades, deve especificar toda a necessidade do usuário.
- c) O sistema é construído em pequenos blocos, que irão compor uma versão a ser entregue aos usuários.
- d) A documentação de projeto deve ser feita pelo próprio desenvolvedor, seguindo padrões simplificados.
- e) Para atingir os objetivos de agilidade exigidos, os desenvolvedores devem seguir processos simplificados para a construção do software.

Na metodologia Scrum, o responsável por definir uma lista de critérios de aceitação do produto é o

- a) chefe do desenvolvimento.
- b) product owner.
- c) scrum master.
- d) time de desenvolvimento.
- e) gerente de orçamento.

Na metodologia Scrum, o responsável por definir uma lista de critérios de aceitação do produto é o

- a) chefe do desenvolvimento.
- b) product owner.
- c) scrum master.
- d) time de desenvolvimento.
- e) gerente de orçamento.

Na elaboração das estórias que irão compor o backlog de um Sprint do Scrum, devem-se levantar os objetivos e ser

- a) mais técnico e propor solução.
- b) menos técnico e desenhar um workflow.
- c) mais técnico e detalhar o desenvolvimento.
- d) menos técnico e se ater ao negócio.
- e) menos técnico e focar nos componentes já existentes.

Na elaboração das estórias que irão compor o backlog de um Sprint do Scrum, devem-se levantar os objetivos e ser

- a) mais técnico e propor solução.
- b) menos técnico e desenhar um workflow.
- c) mais técnico e detalhar o desenvolvimento.
- ➔ d) menos técnico e se ater ao negócio.
- e) menos técnico e focar nos componentes já existentes.

Na reunião de planejamento do Sprint, o backlog do produto a ser desenvolvido é definido apenas pelo(a)

- a) product owner.
- b) equipe de desenvolvimento.
- c) time Scrum.
- d) equipe de infraestrutura.
- e) scrum master.

Na reunião de planejamento do Sprint, o backlog do produto a ser desenvolvido é definido apenas pelo(a)

a) product owner.

→ b) equipe de desenvolvimento.

c) time Scrum.

d) equipe de infraestrutura.

e) scrum master.

O cancelamento do time-box do Sprint, antes do seu término, só poderá ser realizado formalmente pelo

- a) product owner.
- b) administrador de infraestrutura.
- c) scrum master.
- d) designer de interface.
- e) analista de usabilidade.

O cancelamento do time-box do Sprint, antes do seu término, só poderá ser realizado formalmente pelo

- a) product owner.
- b) administrador de infraestrutura.
- c) scrum master.
- d) designer de interface.
- e) analista de usabilidade.

Tendo em vista que, em um processo ágil de desenvolvimento de software, foi adotado o XP (eXtreme Programming) e que os requisitos levantados foram expressos na forma de histórias de usuário, assinale a opção que apresenta, corretamente, recomendações técnicas para a elaboração de um cartão de histórias de usuário.

- a) Como um professor, quero calcular as médias semestrais dos alunos de modo que eu possa identificar quais serão aprovados.
- b) O professor deseja o cálculo de notas semestrais com precisão de até duas casas decimais.
- c) O sistema deve calcular as médias semestrais dos alunos com base nas notas atribuídas a eles pelos professores.
- d) Como analista de requisitos, eu preciso oferecer o cálculo das notas semestrais aos professores em menos de um minuto.
- e) Como um professor, eu preciso de releases semanais de funcionalidades, mesmo que elas possam ser refatoradas posteriormente.

Tendo em vista que, em um processo ágil de desenvolvimento de software, foi adotado o XP (eXtreme Programming) e que os requisitos levantados foram expressos na forma de histórias de usuário, assinale a opção que apresenta, corretamente, recomendações técnicas para a elaboração de um cartão de histórias de usuário.

- a) Como um professor, quero calcular as médias semestrais dos alunos de modo que eu possa identificar quais serão aprovados.
- b) O professor deseja o cálculo de notas semestrais com precisão de até duas casas decimais.
- c) O sistema deve calcular as médias semestrais dos alunos com base nas notas atribuídas a eles pelos professores.
- d) Como analista de requisitos, eu preciso oferecer o cálculo das notas semestrais aos professores em menos de um minuto.
- e) Como um professor, eu preciso de releases semanais de funcionalidades, mesmo que elas possam ser refatoradas posteriormente.

Assinale a opção correta a respeito do emprego do product backlog na abordagem ágil para o levantamento de requisitos e para o desenvolvimento de sistemas, de forma coerente com a interpretação do Scrum.

- a) Um product backlog é uma lista de tarefas que poderão ser desenvolvidas em, aproximadamente, quatro semanas.
- b) Após a descoberta das users histories, os times de desenvolvimento são responsáveis por definir as histórias de maior prioridade do product backlog.
- c) O product backlog deve ser completo ao início do projeto em relação às histórias de usuário, para evitar adaptações no planejamento.
- d) Quando o sistema entra em produção, o product backlog é encerrado.
- e) O product backlog documenta a descrição, a ordem, a estimativa e o valor dos seus itens componentes, reunindo as funções e os requisitos do produto, bem como as melhorias que devem ser feitas nesse produto nas futuras versões.

Assinale a opção correta a respeito do emprego do product backlog na abordagem ágil para o levantamento de requisitos e para o desenvolvimento de sistemas, de forma coerente com a interpretação do Scrum.

- a) Um product backlog é uma lista de tarefas que poderão ser desenvolvidas em, aproximadamente, quatro semanas.
- b) Após a descoberta das users histories, os times de desenvolvimento são responsáveis por definir as histórias de maior prioridade do product backlog.
- c) O product backlog deve ser completo ao início do projeto em relação às histórias de usuário, para evitar adaptações no planejamento.
- d) Quando o sistema entra em produção, o product backlog é encerrado.
- ➔ e) O product backlog documenta a descrição, a ordem, a estimativa e o valor dos seus itens componentes, reunindo as funções e os requisitos do produto, bem como as melhorias que devem ser feitas nesse produto nas futuras versões.

No desenvolvimento ágil de sistemas utilizando o Scrum, um integrante da equipe é encarregado de comunicar a visão, os objetivos e os itens do product backlog para o time de desenvolvimento, além de encontrar técnicas para o gerenciamento efetivo do product backlog. Esse integrante é o

- a) Product Owner, sob orientação do Scrum Master.
- b) próprio time de desenvolvimento, que realiza essas definições de forma auto-organizada.
- c) Scrum Master.
- d) Team Leader.
- e) Product Owner, diretamente.

No desenvolvimento ágil de sistemas utilizando o Scrum, um integrante da equipe é encarregado de comunicar a visão, os objetivos e os itens do product backlog para o time de desenvolvimento, além de encontrar técnicas para o gerenciamento efetivo do product backlog. Esse integrante é o

- a) Product Owner, sob orientação do Scrum Master.
- b) próprio time de desenvolvimento, que realiza essas definições de forma auto-organizada.
- ➔ c) Scrum Master.
- d) Team Leader.
- e) Product Owner, diretamente.

Em um desenvolvimento ágil de sistemas utilizando o XP, foram adotadas as seguintes ações: foi dita a verdade ao cliente acerca do progresso do projeto e acerca de suas estimativas, além de haverem sido realizadas adaptações quando mudanças importantes aconteceram no projeto. Essas ações estão coerentes com o valor do XP denominado

- A) sinceridade.
- B) comunicação.
- C) coragem.
- D) feedback.
- E) respeito.

Em um desenvolvimento ágil de sistemas utilizando o XP, foram adotadas as seguintes ações: foi dita a verdade ao cliente acerca do progresso do projeto e acerca de suas estimativas, além de haverem sido realizadas adaptações quando mudanças importantes aconteceram no projeto. Essas ações estão coerentes com o valor do XP denominado

- A) sinceridade.
- B) comunicação.
- C) coragem.
- D) feedback.
- E) respeito.

O recurso que, devido ao fato de ser considerado guardião do processo, garante que as técnicas de Scrum sejam aplicadas em um projeto denomina-se

- a) líder de equipe.
- b) coordenador do projeto.
- c) líder técnico.
- d) Product Owner.
- e) Scrum Master.

O recurso que, devido ao fato de ser considerado guardião do processo, garante que as técnicas de Scrum sejam aplicadas em um projeto denomina-se

- a) líder de equipe.
- b) coordenador do projeto.
- c) líder técnico.
- d) Product Owner.
- e) Scrum Master.

O desenvolvimento de um software utilizando o Scrum permite que o PO (product owner) ordene os itens do backlog do produto para alcançar melhor as metas e as missões, com o auxílio do SM (Scrum master), na busca de técnicas para o gerenciamento efetivo do backlog do produto, e permite, também, que os desenvolvedores codifiquem os softwares em pares, utilizando-se a prática do XP (extreme programming).

Certo

Errado

O desenvolvimento de um software utilizando o Scrum permite que o PO (product owner) ordene os itens do backlog do produto para alcançar melhor as metas e as missões, com o auxílio do SM (Scrum master), na busca de técnicas para o gerenciamento efetivo do backlog do produto, e permite, também, que os desenvolvedores codifiquem os softwares em pares, utilizando-se a prática do XP (extreme programming).

 Certo

Errado

O SCRUM é uma metodologia ágil para gestão e planejamento de projetos de software. Nele, as funcionalidades a serem implementadas em um projeto são mantidas em uma lista que é conhecida como \_\_\_\_\_. No início de cada Sprint, faz-se um \_\_\_\_\_ na qual o \_\_\_\_\_ prioriza os itens do \_\_\_\_\_ e a equipe seleciona as atividades que ela será capaz de implementar durante o Sprint que se inicia. As tarefas alocadas em um Sprint são transferidas do \_\_\_\_\_ para o \_\_\_\_\_.

- a) Product Backlog, Sprint Backlog, Product Owner, Product Backlog, Sprint Planning Meeting e Product Backlog.
- b) Sprint Planning Meeting, Product Backlog, Product Owner, Product Backlog, Product Backlog e Sprint Backlog.
- c) Product Backlog, Sprint Planning Meeting, Product Owner, Product Backlog, Sprint Backlog e Product Backlog.
- d) Product Backlog, Sprint Planning Meeting, Product Backlog, Sprint Backlog, Product Backlog, e Product Owner.
- e) Product Backlog, Sprint Planning Meeting, Product Owner, Product Backlog, Product Backlog e Sprint Backlog.

O SCRUM é uma metodologia ágil para gestão e planejamento de projetos de software. Nele, as funcionalidades a serem implementadas em um projeto são mantidas em uma lista que é conhecida como \_\_\_\_\_. No início de cada Sprint, faz-se um \_\_\_\_\_ na qual o \_\_\_\_\_ prioriza os itens do \_\_\_\_\_ e a equipe seleciona as atividades que ela será capaz de implementar durante o Sprint que se inicia. As tarefas alocadas em um Sprint são transferidas do \_\_\_\_\_ para o \_\_\_\_\_.

- a) Product Backlog, Sprint Backlog, Product Owner, Product Backlog, Sprint Planning Meeting e Product Backlog.
- b) Sprint Planning Meeting, Product Backlog, Product Owner, Product Backlog, Product Backlog e Sprint Backlog.
- c) Product Backlog, Sprint Planning Meeting, Product Owner, Product Backlog, Sprint Backlog e Product Backlog.
- d) Product Backlog, Sprint Planning Meeting, Product Backlog, Sprint Backlog, Product Backlog, e Product Owner.
- ➔ e) Product Backlog, Sprint Planning Meeting, Product Owner, Product Backlog, Product Backlog e Sprint Backlog.

O SCRUM é um modelo de desenvolvimento ágil de software que fornece métodos para se definir o planejamento, os principais papéis de pessoas e a forma de trabalho da equipe. Em relação ao SCRUM, é correto afirmar que:

- a) o Product Backlog é composto por uma lista de funcionalidades desenvolvida em conjunto com o cliente, que é organizada por prioridade de entrega.
- b) o Proprietário do Produto representa os interesses da equipe, fazendo a interface entre o cliente e o time de desenvolvedores.
- c) o Sprint Backlog é um gráfico que mostra a quantidade de trabalho cumulativo restante de um Sprint, dia por dia.
- d) os três componentes principais do modelo são: papéis, reuniões de planejamento e artefatos.
- e) uma das atribuições do ScrumMaster é assegurar que a equipe de desenvolvimento funcione plenamente, independente da produtividade.

O SCRUM é um modelo de desenvolvimento ágil de software que fornece métodos para se definir o planejamento, os principais papéis de pessoas e a forma de trabalho da equipe. Em relação ao SCRUM, é correto afirmar que:

- ➔ a) o Product Backlog é composto por uma lista de funcionalidades desenvolvida em conjunto com o cliente, que é organizada por prioridade de entrega.
- b) o Proprietário do Produto representa os interesses da equipe, fazendo a interface entre o cliente e o time de desenvolvedores.
- c) o Sprint Backlog é um gráfico que mostra a quantidade de trabalho cumulativo restante de um Sprint, dia por dia.
- d) os três componentes principais do modelo são: papéis, reuniões de planejamento e artefatos.
- e) uma das atribuições do ScrumMaster é assegurar que a equipe de desenvolvimento funcione plenamente, independente da produtividade.

O manifesto ágil tem por princípio que:

- a) mudanças nos requisitos são bem-vindas, mesmo tardiamente no desenvolvimento;
- b) a contínua atenção à excelência técnica reduz a agilidade;
- c) a redução do backlog é a medida primária de progresso;
- d) as melhores arquiteturas, requisitos e designs emergem de equipes que possuem um bom líder;
- e) pessoas de negócio e desenvolvedores devem trabalhar em ambientes separados para reduzir as interferências no processo de desenvolvimento.

O manifesto ágil tem por princípio que:

- ➔ a) mudanças nos requisitos são bem-vindas, mesmo tardiamente no desenvolvimento;
- b) a contínua atenção à excelência técnica reduz a agilidade;
- c) a redução do backlog é a medida primária de progresso;
- d) as melhores arquiteturas, requisitos e designs emergem de equipes que possuem um bom líder;
- e) pessoas de negócio e desenvolvedores devem trabalhar em ambientes separados para reduzir as interferências no processo de desenvolvimento.

## Questão 16 - FGV - 2015 – TCE/SE

As metodologias ágeis representam um conjunto estruturado de práticas para o desenvolvimento de projetos de sistemas de software de forma iterativa e incremental. Em relação às metodologias ágeis, analise as afirmativas a seguir:

- I. No Scrum, se um ou mais aspectos de um processo desviou para fora dos limites aceitáveis, e é percebido que o produto resultado será inaceitável, o processo ou o material sendo produzido deve ser registrado para ser ajustado na próxima iteração.
- II. Desenvolvimento Guiado por Funcionalidades (FDD) é uma metodologia ágil que se destaca pelo fato de entregar, mensalmente, blocos bem pequenos de funcionalidades valorizadas pelo cliente. No entanto, o monitoramento dentro do projeto não é detalhado.
- III. O desenvolvimento orientado a testes é uma prática recomendada pela metodologia de desenvolvimento de software XP. Escrever os testes antes de construir o sistema torna mais fácil entender como o código deve ser programado.

Está correto o que se afirma em:

- a) somente I;
- b) somente II;
- c) somente III;
- d) somente I e III;
- e) I, II e III.

## Questão 16 - FGV - 2015 – TCE/SE

As metodologias ágeis representam um conjunto estruturado de práticas para o desenvolvimento de projetos de sistemas de software de forma iterativa e incremental. Em relação às metodologias ágeis, analise as afirmativas a seguir:

- I. No Scrum, se um ou mais aspectos de um processo desviou para fora dos limites aceitáveis, e é percebido que o produto resultado será inaceitável, o processo ou o material sendo produzido deve ser registrado para ser ajustado na próxima iteração.
- II. Desenvolvimento Guiado por Funcionalidades (FDD) é uma metodologia ágil que se destaca pelo fato de entregar, mensalmente, blocos bem pequenos de funcionalidades valorizadas pelo cliente. No entanto, o monitoramento dentro do projeto não é detalhado.
- III. O desenvolvimento orientado a testes é uma prática recomendada pela metodologia de desenvolvimento de software XP. Escrever os testes antes de construir o sistema torna mais fácil entender como o código deve ser programado.

Está correto o que se afirma em:

- a) somente I;
- b) somente II;
-  c) somente III;
- d) somente I e III;
- e) I, II e III.

O processo de escrever testes antes da implementação de um projeto de software é conhecido como Test-Driven Development (TDD).

A respeito do processo de desenvolvimento dirigido por testes, assinale a afirmativa correta.

- a) Testes ajudam na modelagem do projeto de software guiando o desenvolvedor no desenho das classes.
- b) Após a implementação do projeto de software, os componentes de testes devem ser descartados porque não se tornarão parte da manutenção do projeto.
- c) Não há necessidade de refatoração de código, uma vez que essa técnica de desenvolvimento pressupõe que o código será produzido em padrões aceitáveis.
- d) O desenvolvedor precisa escrever casos de teste automatizados que vão além das fronteiras do projeto de software para validar o comportamento esperado em processos externos.
- e) Testes de unidade automatizados que definam requisitos em código são criados somente após escrever o código da aplicação.

O processo de escrever testes antes da implementação de um projeto de software é conhecido como Test-Driven Development (TDD).

A respeito do processo de desenvolvimento dirigido por testes, assinale a afirmativa correta.

- ➔ a) Testes ajudam na modelagem do projeto de software guiando o desenvolvedor no desenho das classes.
- b) Após a implementação do projeto de software, os componentes de testes devem ser descartados porque não se tornarão parte da manutenção do projeto.
- c) Não há necessidade de refatoração de código, uma vez que essa técnica de desenvolvimento pressupõe que o código será produzido em padrões aceitáveis.
- d) O desenvolvedor precisa escrever casos de teste automatizados que vão além das fronteiras do projeto de software para validar o comportamento esperado em processos externos.
- e) Testes de unidade automatizados que definam requisitos em código são criados somente após escrever o código da aplicação.

## Questão 18 - FGV - 2014 – PROCEMPA

Com relação às metodologias de desenvolvimento de projetos de software, analise as afirmativas a seguir:

- I. Scrum permite a construção de software incrementalmente por meio de iterações curtas para promover visibilidade para o desenvolvimento e pressupõem equipes pequenas, requisitos pouco estáveis ou desconhecidos.
- II. Feature Driven Development (FDD) suporta o desenvolvimento ágil com rápidas adaptações às mudanças de requisitos focados nas fases de desenho e construção de projeto de software.
- III. Kanban considera a utilização de uma sinalização ou registro visual para gerenciar o limite de atividades em andamento, indicando se um novo trabalho pode ou não ser iniciado e se o limite acordado para cada fase está sendo respeitado.

Assinale:

- a) se somente a afirmativa I estiver correta.
- b) se somente a afirmativa II estiver correta.
- c) se somente a afirmativa III estiver correta.
- d) se somente as afirmativas I e II estiverem corretas.
- e) se todas as afirmativas estiverem corretas.

## Questão 18 - FGV - 2014 – PROCEMPA

Com relação às metodologias de desenvolvimento de projetos de software, analise as afirmativas a seguir:

- I. Scrum permite a construção de software incrementalmente por meio de iterações curtas para promover visibilidade para o desenvolvimento e pressupõem equipes pequenas, requisitos pouco estáveis ou desconhecidos.
- II. Feature Driven Development (FDD) suporta o desenvolvimento ágil com rápidas adaptações às mudanças de requisitos focados nas fases de desenho e construção de projeto de software.
- III. Kanban considera a utilização de uma sinalização ou registro visual para gerenciar o limite de atividades em andamento, indicando se um novo trabalho pode ou não ser iniciado e se o limite acordado para cada fase está sendo respeitado.

Assinale:

- a) se somente a afirmativa I estiver correta.
- b) se somente a afirmativa II estiver correta.
- c) se somente a afirmativa III estiver correta.
- d) se somente as afirmativas I e II estiverem corretas.
- e) se todas as afirmativas estiverem corretas.

No método XP (eXtreming programming), os sistemas são concebidos a partir de uma metáfora e descritos em estórias do usuário. Esse método busca facilitar a comunicação com o cliente, entendendo a realidade deste e guiando o desenvolvimento com o uso de estória simples.

Certo

Errado

No método XP (eXtreming programming), os sistemas são concebidos a partir de uma metáfora e descritos em estórias do usuário. Esse método busca facilitar a comunicação com o cliente, entendendo a realidade deste e guiando o desenvolvimento com o uso de estória simples.

→ Certo

Errado

Dentro de uma Sprint no Scrum, as metas de qualidade não diminuem e não são feitas mudanças que possam afetar o objetivo da Sprint.

Certo

Errado

Dentro de uma Sprint no Scrum, as metas de qualidade não diminuem e não são feitas mudanças que possam afetar o objetivo da Sprint.

→ Certo

Errado

No SCRUM, cada ponto de história (PH) implica uma hora de trabalho de uma pessoa.

Certo

Errado

No SCRUM, cada ponto de história (PH) implica uma hora de trabalho de uma pessoa.

Certo



Errado

O Scrummaster deve assumir a gerência de um projeto ágil com base no SCRUM, de modo a definir as prioridades para que a equipe entregue, primeiramente, os produtos de software que agreguem maior valor ao negócio do cliente.

Certo

Errado

O Scrummaster deve assumir a gerência de um projeto ágil com base no SCRUM, de modo a definir as prioridades para que a equipe entregue, primeiramente, os produtos de software que agreguem maior valor ao negócio do cliente.

Certo

→ Errado

A etapa de planejamento do Extreme Programming (XP) inicia-se com a escrita de UserStories (história do usuário). Por meio dessa ferramenta, aqueles que conhecem a técnica de construção de uma solução poderão guiar quem necessita dessa solução no exercício de descrevê-la de forma simples e concisa.

Certo

Errado

A etapa de planejamento do Extreme Programming (XP) inicia-se com a escrita de UserStories (história do usuário). Por meio dessa ferramenta, aqueles que conhecem a técnica de construção de uma solução poderão guiar quem necessita dessa solução no exercício de descrevê-la de forma simples e concisa.

→ Certo

Errado

No Scrum, um projeto se inicia com uma visão simples do produto que será desenvolvido. A visão pode ser vaga a princípio e ir tornando-se clara aos poucos. O I então, transforma essa visão em uma lista de requisitos funcionais e não-funcionais para que, quando forem desenvolvidos, reflitam essa visão. Essa lista, chamada de II, é priorizada pelo III de forma que os itens que gerem maior valor ao produto tenham maior prioridade.

Completa, correta e respectivamente, as lacunas I, II e III:

- a) Daily Scrum - Scrum Team - Sprint
- b) Daily Scrum - Product Backlog - Sprint Planning Meeting
- c) Product Owner - Sprint - Product Backlog
- d) Scrum Team - Sprint Planning Meeting - Product Owner
- e) Product Owner - Product Backlog - Product Owner

No Scrum, um projeto se inicia com uma visão simples do produto que será desenvolvido. A visão pode ser vaga a princípio e ir tornando-se clara aos poucos. O I então, transforma essa visão em uma lista de requisitos funcionais e não-funcionais para que, quando forem desenvolvidos, reflitam essa visão. Essa lista, chamada de II, é priorizada pelo III de forma que os itens que gerem maior valor ao produto tenham maior prioridade.

Completa, correta e respectivamente, as lacunas I, II e III:

- a) Daily Scrum - Scrum Team - Sprint
- b) Daily Scrum - Product Backlog - Sprint Planning Meeting
- c) Product Owner - Sprint - Product Backlog
- d) Scrum Team - Sprint Planning Meeting - Product Owner
- ➔ e) Product Owner - Product Backlog - Product Owner

No Scrum, segundo o guia 2013, o responsável pelo trabalho de expressar claramente os itens do Backlog do Produto é o

- a) Product Master
- b) Product Owner
- c) Scrum Master
- d) Scrum Owner
- e) Time de Desenvolvimento

No Scrum, segundo o guia 2013, o responsável pelo trabalho de expressar claramente os itens do Backlog do Produto é o

a) Product Master

→ b) Product Owner

c) Scrum Master

d) Scrum Owner

e) Time de Desenvolvimento

Acerca do SCRUM, assinale a opção correta.

- a) Entre os objetivos da retrospectiva da sprint, inclui-se avaliar o desempenho de cada integrante da equipe de desenvolvimento. A retrospectiva da Sprint deve ocorrer antes da revisão da Sprint, antes da reunião de planejamento da próxima sprint e depois do término da sprint.
- b) Uma vez que as sprints no SCRUM são fixadas em uma time-box de um mês, as reuniões de planejamento das sprints seguem uma time-box de quatro horas.
- c) A reunião diária do SCRUM é um evento time-boxed de quinze minutos realizado para determinar o trabalho que deverá ser feito antes da próxima reunião diária com a participação do Scrum Master e do Product Owner.
- d) Como o cancelamento de uma sprint é um evento indesejado no SCRUM, esse evento só ocorre se o Scrum Master constatar que os objetivos pretendidos para tal Sprint são inalcançáveis.
- e) O Product Owner utiliza práticas de estimativa como oburndown e o burnup para acompanhar a quantidade de trabalho que a organização ainda deve realizar para alcançar os seus objetivos. Assim, se o propósito for garantir o alcance de objetivos, o trabalho poderá ser resumido em qualquer ponto do tempo.

Acerca do SCRUM, assinale a opção correta.

- a) Entre os objetivos da retrospectiva da sprint, inclui-se avaliar o desempenho de cada integrante da equipe de desenvolvimento. A retrospectiva da Sprint deve ocorrer antes da revisão da Sprint, antes da reunião de planejamento da próxima sprint e depois do término da sprint.
- b) Uma vez que as sprints no SCRUM são fixadas em uma time-box de um mês, as reuniões de planejamento das sprints seguem uma time-box de quatro horas.
- c) A reunião diária do SCRUM é um evento time-boxed de quinze minutos realizado para determinar o trabalho que deverá ser feito antes da próxima reunião diária com a participação do Scrum Master e do Product Owner.
- d) Como o cancelamento de uma sprint é um evento indesejado no SCRUM, esse evento só ocorre se o Scrum Master constatar que os objetivos pretendidos para tal Sprint são inalcançáveis.
- e) O Product Owner utiliza práticas de estimativa como oburndown e o burnup para acompanhar a quantidade de trabalho que a organização ainda deve realizar para alcançar os seus objetivos. Assim, se o propósito for garantir o alcance de objetivos, o trabalho poderá ser resumido em qualquer ponto do tempo.

No TDD, o refatoramento do código deve ser realizado antes de se escrever a aplicação que deve ser testada.

Certo

Errado

No TDD, o refatoramento do código deve ser realizado antes de se escrever a aplicação que deve ser testada.

Certo

→ Errado

No planejamento de projetos de software, e principalmente em metodologias ágeis de desenvolvimento, muitos autores defendem a técnica conhecida como “timebox”, que:

- a) estima o menor e o maior tempo de desenvolvimento para cada funcionalidade a ser desenvolvida, definindo uma “caixa” de tempo em vez de um prazo fixo.
- b) parte do tempo disponível em uma fábrica de software para especificar versões consecutivas de um produto, conhecidas como “caixas”
- c) divide um produto de software em versões de complexidade crescente, conhecidas como “caixas”, especificando o tempo de desenvolvimento de cada caixa do mais rápido para o mais longo.
- d) define um tempo para cada função a ser desenvolvida e as aloca em “caixas” de igual tempo de desenvolvimento que são escolhidas pelos desenvolvedores.
- e) define o tempo a ser utilizado em um ciclo de desenvolvimento e depois define a funcionalidade que pode ser desenvolvida naquela “caixa” de tempo.

No planejamento de projetos de software, e principalmente em metodologias ágeis de desenvolvimento, muitos autores defendem a técnica conhecida como “timebox”, que:

- a) estima o menor e o maior tempo de desenvolvimento para cada funcionalidade a ser desenvolvida, definindo uma “caixa” de tempo em vez de um prazo fixo.
- b) parte do tempo disponível em uma fábrica de software para especificar versões consecutivas de um produto, conhecidas como “caixas”
- c) divide um produto de software em versões de complexidade crescente, conhecidas como “caixas”, especificando o tempo de desenvolvimento de cada caixa do mais rápido para o mais longo.
- d) define um tempo para cada função a ser desenvolvida e as aloca em “caixas” de igual tempo de desenvolvimento que são escolhidas pelos desenvolvedores.
- e) define o tempo a ser utilizado em um ciclo de desenvolvimento e depois define a funcionalidade que pode ser desenvolvida naquela “caixa” de tempo.

## Questão 29 - FCC - 2014 – AL/PE

O Scrum define reuniões e eventos que devem ser realizados de forma a oferecer oportunidades formais para inspeção e adaptação, cujos tempos de duração são referenciais máximos recomendados. Considere:

- I. É uma Sprint de um mês, para inspecionar o incremento e adaptar o Backlog do Produto, se necessário.
- II. É uma reunião time-boxed de 3 horas para uma Sprint de um mês, sendo uma oportunidade para o Time Scrum inspecionar a si próprio e criar um plano para melhorias a serem aplicadas na próxima Sprint.
- III. É um evento time-boxed de 15 minutos, para que a Equipe de Desenvolvimento possa sincronizar as atividades e criar um plano para as próximas 24 horas.
- IV. É um time-box de 8 horas para uma Sprint de um mês de duração.

Estão de acordo com as definições I, II, III e IV, respectivamente, as denominações:

- a) planejamento da Sprint - revisão da Sprint - daily Scrum - retrospectiva da Sprint
- b) revisão da Sprint - retrospectiva da Sprint - daily Scrum - planejamento da Sprint
- c) revisão da Sprint - planejamento da Sprint - 15 min break - retrospectiva da Sprint
- d) retrospectiva da Sprint - planejamento da Sprint - short meeting - revisão da Sprint
- e) planejamento da Sprint - retrospectiva da Sprint - daily Scrum - revisão da Sprint

## Questão 29 - FCC - 2014 – AL/PE

O Scrum define reuniões e eventos que devem ser realizados de forma a oferecer oportunidades formais para inspeção e adaptação, cujos tempos de duração são referenciais máximos recomendados. Considere:

- I. É uma Sprint de um mês, para inspecionar o incremento e adaptar o Backlog do Produto, se necessário.
- II. É uma reunião time-boxed de 3 horas para uma Sprint de um mês, sendo uma oportunidade para o Time Scrum inspecionar a si próprio e criar um plano para melhorias a serem aplicadas na próxima Sprint.
- III. É um evento time-boxed de 15 minutos, para que a Equipe de Desenvolvimento possa sincronizar as atividades e criar um plano para as próximas 24 horas.
- IV. É um time-box de 8 horas para uma Sprint de um mês de duração.

Estão de acordo com as definições I, II, III e IV, respectivamente, as denominações:

- a) planejamento da Sprint - revisão da Sprint - daily Scrum - retrospectiva da Sprint
- ➔ b) revisão da Sprint - retrospectiva da Sprint - daily Scrum - planejamento da Sprint
- c) revisão da Sprint - planejamento da Sprint - 15 min break - retrospectiva da Sprint
- d) retrospectiva da Sprint - planejamento da Sprint - short meeting - revisão da Sprint
- e) planejamento da Sprint - retrospectiva da Sprint - daily Scrum - revisão da Sprint

## Questão 30 - FCC - 2014 – AL/PE

Scrum e XP são duas metodologias ágeis que provêm práticas e regras que apresentam diferenças e também pontos em comum. Comparando-se estas metodologias, é correto afirmar:

- a) A XP enfatiza a proximidade física do cliente com a equipe de desenvolvimento para facilitar a comunicação. No Scrum existem diversos eventos formais, tais como sprint backlog meeting e product backlog review, que incentivam a comunicação entre todos os profissionais envolvidos no projeto.
- b) As duas metodologias utilizam iterações curtas de desenvolvimento (sprints), mas divergem no tempo de duração das mesmas. Enquanto no Scrum uma sprint dura de 15 minutos a 8 horas, na XP costuma durar de 1 a 24 horas.
- c) Tanto o Scrum quanto a XP explicitamente não permitem que ocorram mudanças de escopo ou definição dentro de uma sprint. Por isso o cliente deve validar todos os requisitos no início do projeto, isso vai contribuir para evitar atrasos e até mesmo construções erradas.
- d) A XP enfatiza que não se deve fazer horas extras constantemente e, se isso ocorrer, existem problemas no projeto que devem ser resolvidos não com aumento de horas, mas com melhor planejamento. O Scrum enfatiza que equipes auto-organizáveis escolhem qual a melhor forma para completarem seu trabalho.
- e) O Scrum estabelece que os testes devem ocorrer o tempo todo durante o desenvolvimento, principalmente usando técnicas automatizadas. Na XP os testes podem ser realizados apenas na parte final de cada sprint, usando a técnica de refatoração, que busca validar todas as funcionalidades, pensando estrategicamente em como refatorar o código que está sendo implementado.

## Questão 30 - FCC - 2014 – AL/PE

Scrum e XP são duas metodologias ágeis que provêm práticas e regras que apresentam diferenças e também pontos em comum. Comparando-se estas metodologias, é correto afirmar:

- a) A XP enfatiza a proximidade física do cliente com a equipe de desenvolvimento para facilitar a comunicação. No Scrum existem diversos eventos formais, tais como sprint backlog meeting e product backlog review, que incentivam a comunicação entre todos os profissionais envolvidos no projeto.
- b) As duas metodologias utilizam iterações curtas de desenvolvimento (sprints), mas divergem no tempo de duração das mesmas. Enquanto no Scrum uma sprint dura de 15 minutos a 8 horas, na XP costuma durar de 1 a 24 horas.
- c) Tanto o Scrum quanto a XP explicitamente não permitem que ocorram mudanças de escopo ou definição dentro de uma sprint. Por isso o cliente deve validar todos os requisitos no início do projeto, isso vai contribuir para evitar atrasos e até mesmo construções erradas.
-  d) A XP enfatiza que não se deve fazer horas extras constantemente e, se isso ocorrer, existem problemas no projeto que devem ser resolvidos não com aumento de horas, mas com melhor planejamento. O Scrum enfatiza que equipes auto-organizáveis escolhem qual a melhor forma para completarem seu trabalho.
- e) O Scrum estabelece que os testes devem ocorrer o tempo todo durante o desenvolvimento, principalmente usando técnicas automatizadas. Na XP os testes podem ser realizados apenas na parte final de cada sprint, usando a técnica de refatoração, que busca validar todas as funcionalidades, pensando estrategicamente em como refatorar o código que está sendo implementado.

## Questão 31 - FCC - 2014 – TRT2

Há diversos processos e práticas ágeis de desenvolvimento de software. Considere:

I. Seu objetivo é criar um “código limpo que funcione”. Trabalha com a estratégia Red - Green - Refactor:

- Codifique o teste;
- Faça-o compilar e executar. O teste não deve passar (Red).
- Implemente o requisito e faça o teste passar (Green).
- Refatore o código (Refactor).

II. Suas práticas, regras e valores garantem um agradável ambiente de desenvolvimento de software para os seus seguidores, que são conduzidos pelos princípios básicos:

- Comunicação - manter o melhor relacionamento possível entre clientes e desenvolvedores, preferindo conversas pessoais a outros meios de comunicação;
- Simplicidade - implementar apenas requisitos atuais, evitando adicionar funcionalidades que podem ser importantes somente no futuro;
- Feedback - o desenvolvedor terá informações constantes do cliente e do código, em que testes constantes indicam os erros tanto individuais quanto do software integrado;
- Coragem - encorajar as pessoas que não possuem facilidade de comunicação e bom relacionamento interpessoal, encorajar a equipe a experimentar e buscar novas soluções, além de encorajar a obtenção de feedback do cliente.

III. Objetiva capturar os critérios de aceitação para as funcionalidades em desenvolvimento. Trabalha com as seguintes etapas:

- Discutir (Discuss): discussão colaborativa com a equipe visando elicitar os critérios de aceitação.
- Refinar (Distill): refinamento dos critérios de aceitação em um conjunto concreto de cenários/exemplos de uso descrevendo o comportamento esperado da aplicação em uma linguagem comum a todos os membros da equipe.
- Desenvolver (Develop): transformação dos testes de aceitação (descrevendo o comportamento esperado do software) em testes/especificação automatizados.

IV. Suas práticas incluem:

- Envolver as partes interessadas no processo através de Outside-in Development.
- Usar exemplos para descrever o comportamento de uma aplicação ou unidades de código.
- Automatizar os exemplos para prover um feedback rápido e testes de regressão.
- Usar o verbo deve (should) ao descrever o comportamento de software para ajudar a esclarecer responsabilidades e permitir que funcionalidades sejam questionadas.
- Usar duplês de teste (mocks, stubs, fakes, dummies, spies) para auxiliar na colaboração entre módulos e códigos que ainda não foram escritos.

Os processos ágeis I, II, III e IV são, correta e respectivamente, denominados:

- a) BDD - DDD - ATDD - XP
- b) TDD - BDD - DDD - XP
- c) ATDD - XP - DDD - BDD
- d) ATDD - BDD - TDD - DDD
- e) TDD - XP - ATDD - BDD

## Questão 31 - FCC - 2014 – TRT2

Há diversos processos e práticas ágeis de desenvolvimento de software. Considere:

I. Seu objetivo é criar um “código limpo que funcione”. Trabalha com a estratégia Red - Green - Refactor:

- Codifique o teste;
- Faça-o compilar e executar. O teste não deve passar (Red).
- Implemente o requisito e faça o teste passar (Green).
- Refatore o código (Refactor).

II. Suas práticas, regras e valores garantem um agradável ambiente de desenvolvimento de software para os seus seguidores, que são conduzidos pelos princípios básicos:

- Comunicação - manter o melhor relacionamento possível entre clientes e desenvolvedores, preferindo conversas pessoais a outros meios de comunicação;
- Simplicidade - implementar apenas requisitos atuais, evitando adicionar funcionalidades que podem ser importantes somente no futuro;
- Feedback - o desenvolvedor terá informações constantes do cliente e do código, em que testes constantes indicam os erros tanto individuais quanto do software integrado;
- Coragem - encorajar as pessoas que não possuem facilidade de comunicação e bom relacionamento interpessoal, encorajar a equipe a experimentar e buscar novas soluções, além de encorajar a obtenção de feedback do cliente.

III. Objetiva capturar os critérios de aceitação para as funcionalidades em desenvolvimento. Trabalha com as seguintes etapas:

- Discutir (Discuss): discussão colaborativa com a equipe visando elicitar os critérios de aceitação.
- Refinar (Distill): refinamento dos critérios de aceitação em um conjunto concreto de cenários/exemplos de uso descrevendo o comportamento esperado da aplicação em uma linguagem comum a todos os membros da equipe.
- Desenvolver (Develop): transformação dos testes de aceitação (descrevendo o comportamento esperado do software) em testes/especificação automatizados.

IV. Suas práticas incluem:

- Envolver as partes interessadas no processo através de Outside-in Development.
- Usar exemplos para descrever o comportamento de uma aplicação ou unidades de código.
- Automatizar os exemplos para prover um feedback rápido e testes de regressão.
- Usar o verbo deve (should) ao descrever o comportamento de software para ajudar a esclarecer responsabilidades e permitir que funcionalidades sejam questionadas.
- Usar duplões de teste (mocks, stubs, fakes, dummies, spies) para auxiliar na colaboração entre módulos e códigos que ainda não foram escritos.

Os processos ágeis I, II, III e IV são, correta e respectivamente, denominados:

- a) BDD - DDD - ATDD - XP
- b) TDD - BDD - DDD - XP
- c) ATDD - XP - DDD - BDD
- d) ATDD - BDD - TDD - DDD
- e) TDD - XP - ATDD - BDD

# Gabarito – Questões Comentadas

1 – B	12 – CERTO	23 – CERTO
2 – A	13 – E	24 – E
3 – B	14 – A	25 – B
4 – D	15 – A	26 – E
5 – B	16 – D	27 – ERRADO
6 – A	17 – A	28 – E
7 – A	18 – E	29 – B
8 – E	19 – CERTO	30 – D
9 – C	20 – CERTO	31 – E
10 – C	21 – ERRADO	
11 – E	22 - ERRADO	