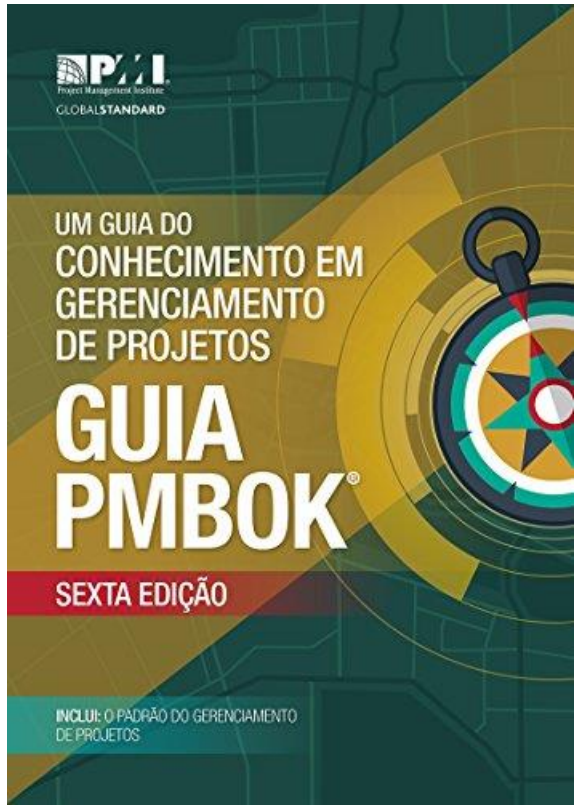
Gerenciamento de Projetos pelo PMBoK 6ª.Edição

GESTÃO DE PROJETOS EM SEGURANÇA DA INFORMAÇÃO

- ✓ Fundamentos do Gerenciamento de Projetos
- ✓ Gerenciamento da Integração
- ✓ Perfil do Gerente de Projetos
- ✓ Partes Interessadas em um projeto
- ✓ Abordagens Ágeis
- Tipos de organização e Influências externas
- Processos de Gerenciamento dos Projetos
- Gerenciamento da Integração
- Gerenciamento do Escopo
- Gerenciamento do Tempo
- Gerenciamento de Custos
- Gerenciamento da Qualidade
- Gerenciamento de Recursos Humanos
- Gerenciamento das Comunicações
- Gerenciamento de Riscos
- Gerenciamento de Aquisições
- Encerramento dos Projetos



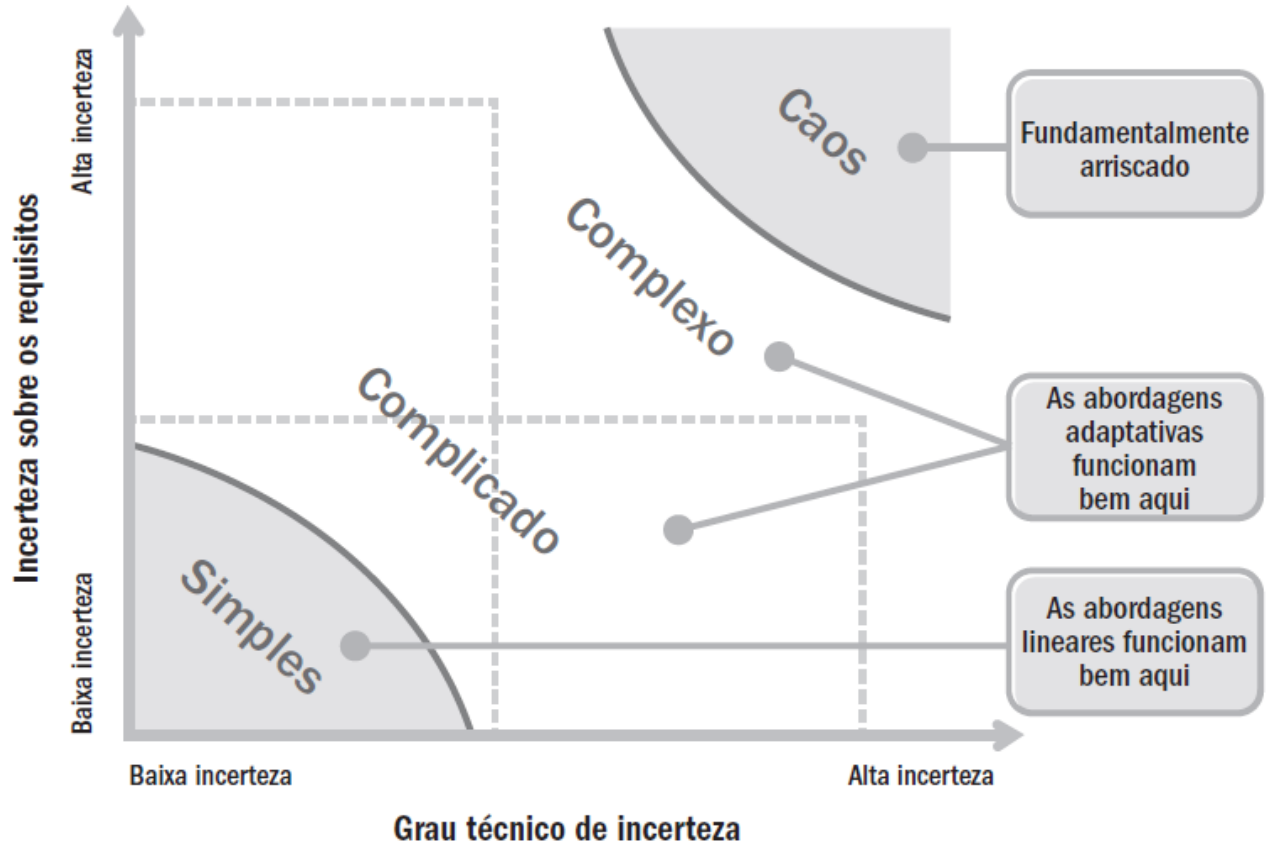
LINK PARA OS QUESTINÁRIOS

- AULA 1 E 2
  - https://forms.gle/617AD2zVFgUr954Q6

ÁBORADAGEM ÁGIL

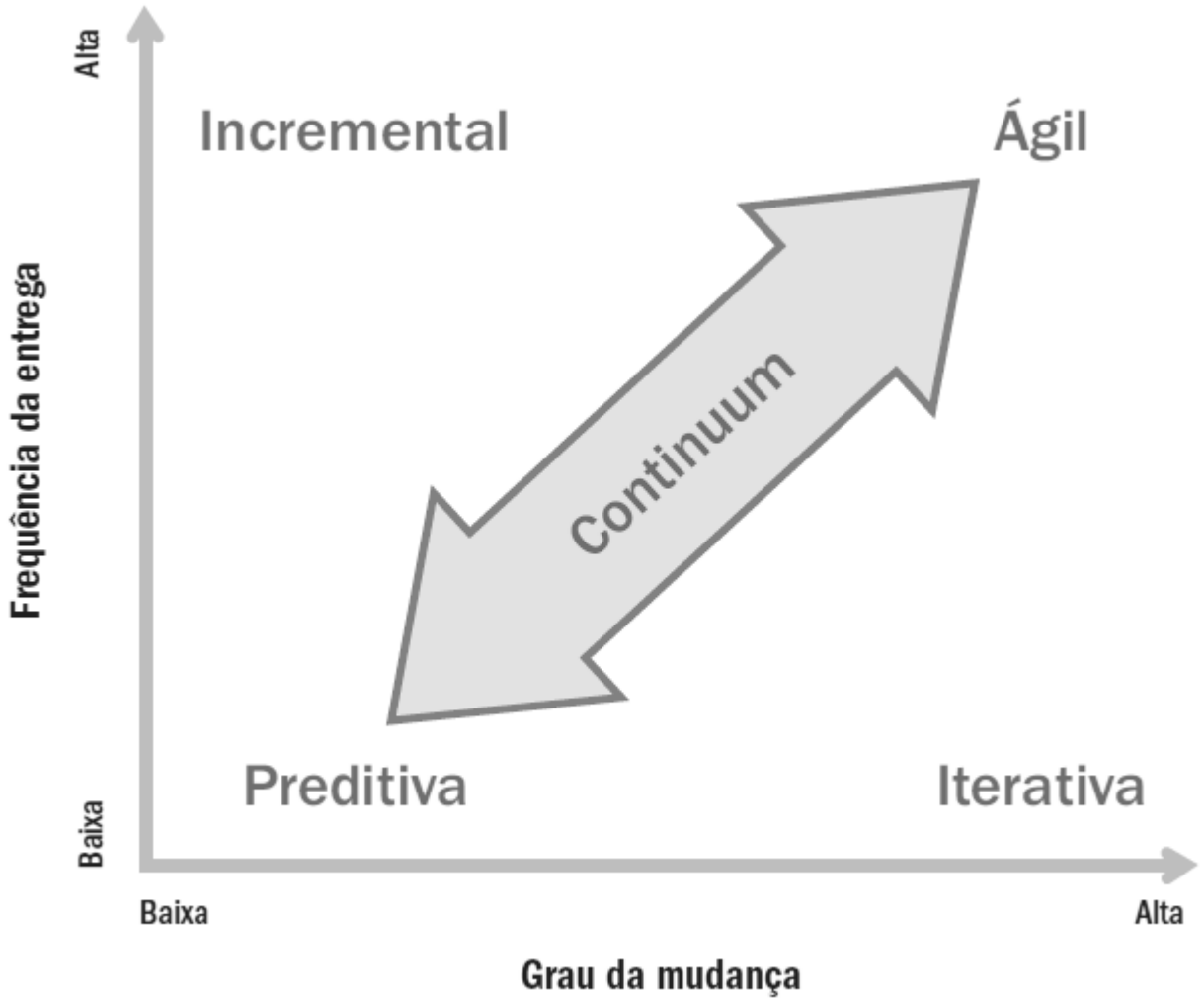
- VAMOS FALAR UM POUCO DAS ABORADAGENS ÁGEIS
- NA SEXTA EDIÇÃO O PMBOOK ADICIONOU ESTES ELEMENTOS NUM CÁPITULO EXTRA
- NO MERCADO DE TI ESSAS ABORDAGENS TEM GANHO MUITO ESPAÇO
- ESTE É UM TÓPICO ADICIONAL DE NOSSO PORGRAMA

Modelo da Complexidade de Stacey



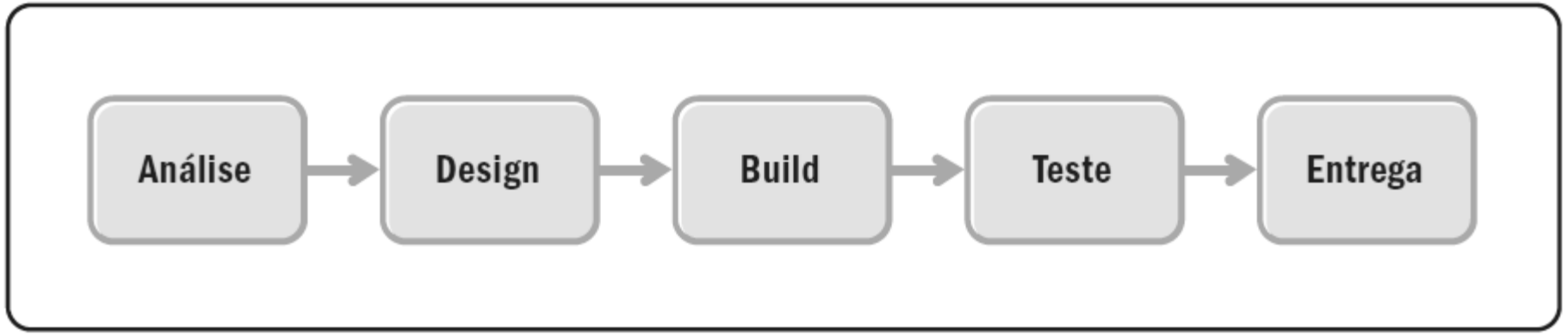
Abordagens de Ciclo de Vida

Características				
Abordagem	**Requisitos**	**Atividades**	**Entrega**	**Objetivo**
Preditivo	Fixo	Realizado uma vez para todo o projeto	Entrega única	Gerenciar o custo
Iterativo	Dinâmico	Repetido até estar correto	Entrega única	Correção da solução
Incremental	Dinâmico	Realizado uma vez para determinado incremento	Entregas menores frequentes	Velocidade
Ágil	Dinâmico	Repetido até estar correto	Entregas menores frequentes	Valor do cliente por meio de entregas e feedback frequentes


Alta
Incremental
Ágil
Continuum
Preditiva
Iterativa
Baixa
Frequência da entrega
Baixa
Alta
Grau da mudança

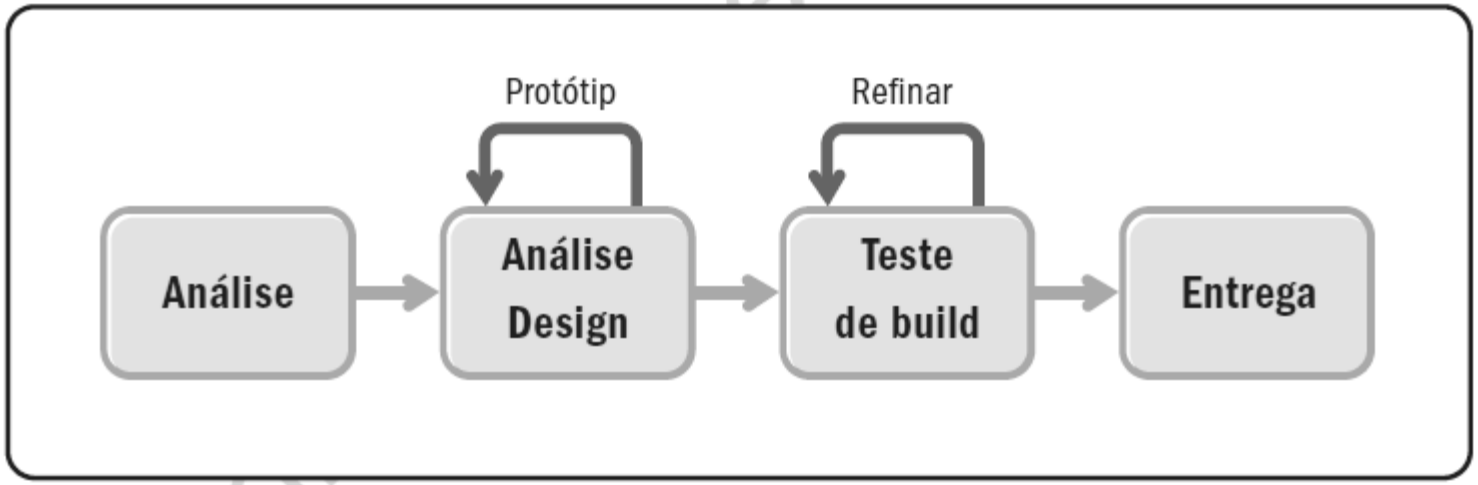
Abordagem preditiva

Os ciclos de vida preditivos almejam aproveitar a alta certeza referente a requisitos , uma equipe estável e o baixo risco. Como resultado, as atividades do projeto geralmente são executadas de forma sequencial,



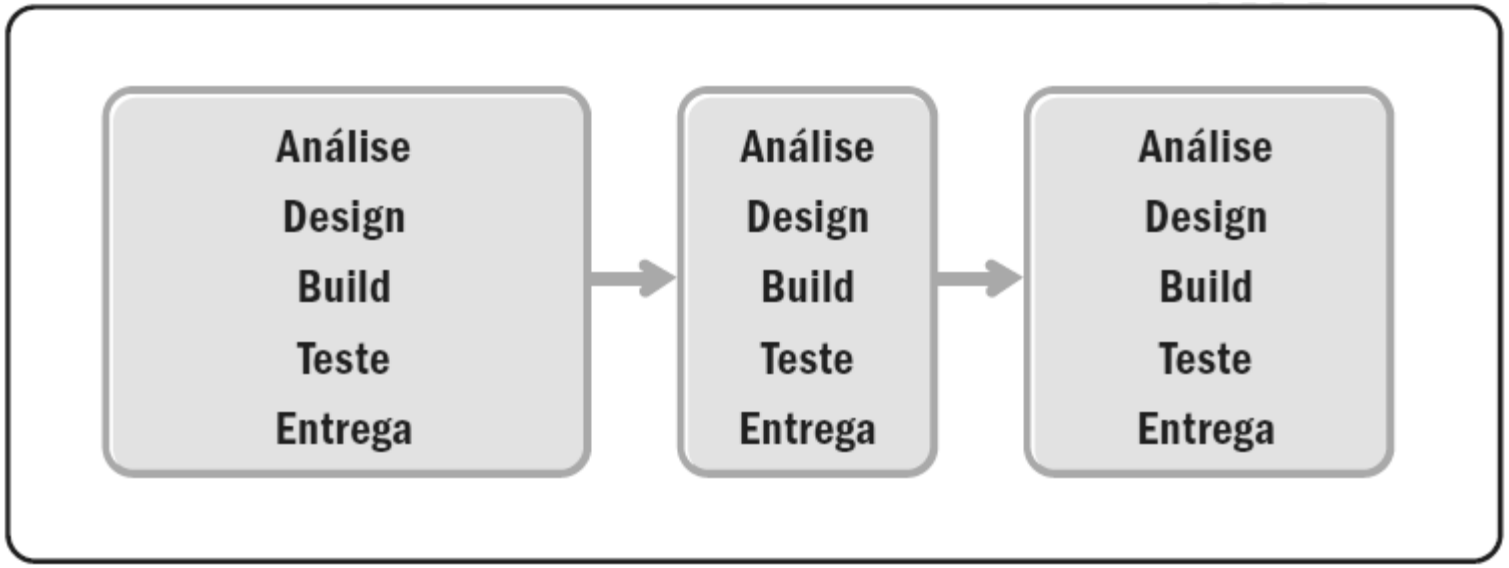
Abordagem iterativa

- Os ciclos iterativos buscam melhores resultados por meio de protótipos
- Recebem feedback do cliente e permitem um aumento evolutivo do conhecimento da equipe
- Eliminam incertezas por meio de prototipação
- Nas fases finais quando a incerteza é menor a solução é refinada por meio de mudanças menores



Abordagem incremental

Alguns projetos otimizam a velocidade de entrega. Muitos negócios e iniciativas não podem esperar que tudo seja concluído; nesses casos, os clientes estão dispostos a receber um subconjunto da solução total. Esse esquema de entregáveis menores frequentes é chamado de ciclo de vida incremental



Abordagem ágil

Ágil Baseado em Iteração

Requisitos Análise Design Build Teste	Requisitos Análise Design Build Teste	Requisitos Análise Design Build Teste	Requisitos Análise Design Build Teste	Repetir conforme necessário ...	Requisitos Análise Design Build Teste	Requisitos Análise Design Build Teste

OBSERVAÇÃO: Cada janela de tempo tem o mesmo tamanho. Cada janela de tempo resulta em trabalhar as funcionalidades testadas.

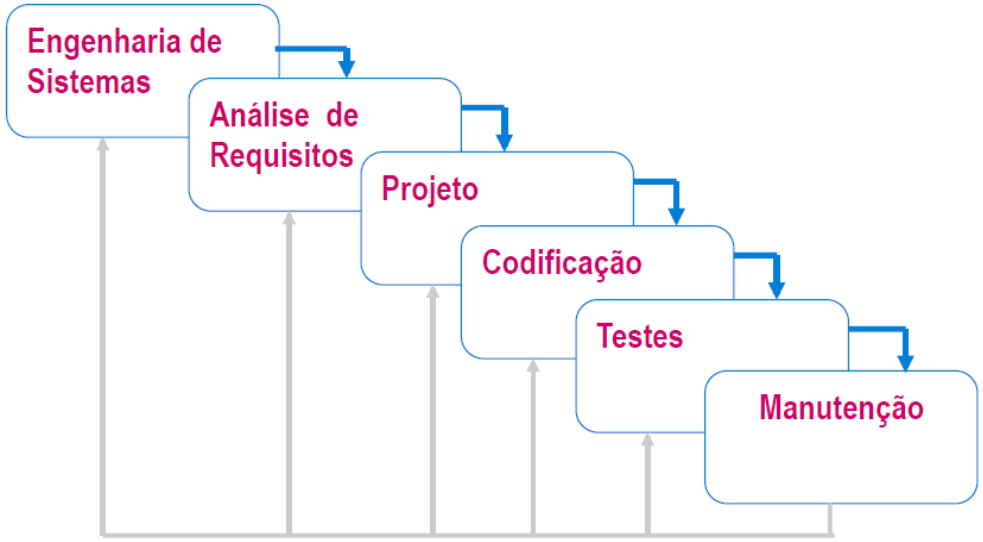
Ágil Baseado em Fluxo

Requisitos Análise Design Build Teste o número de funcionalidades no limite do WIP	Requisitos Análise Design Build Teste o número de funcionali-dades no limite do WIP	Requisitos Análise Design Build Teste o número de funcionalidades no limite do WIP	Repetir conforme necessário ...	Requisitos Análise Design Build Teste o número de funcionalidades no limite do WIP	Requisitos Análise Design Build Teste o número de funcionalidades no limite do WIP

Estrutura de Equipes

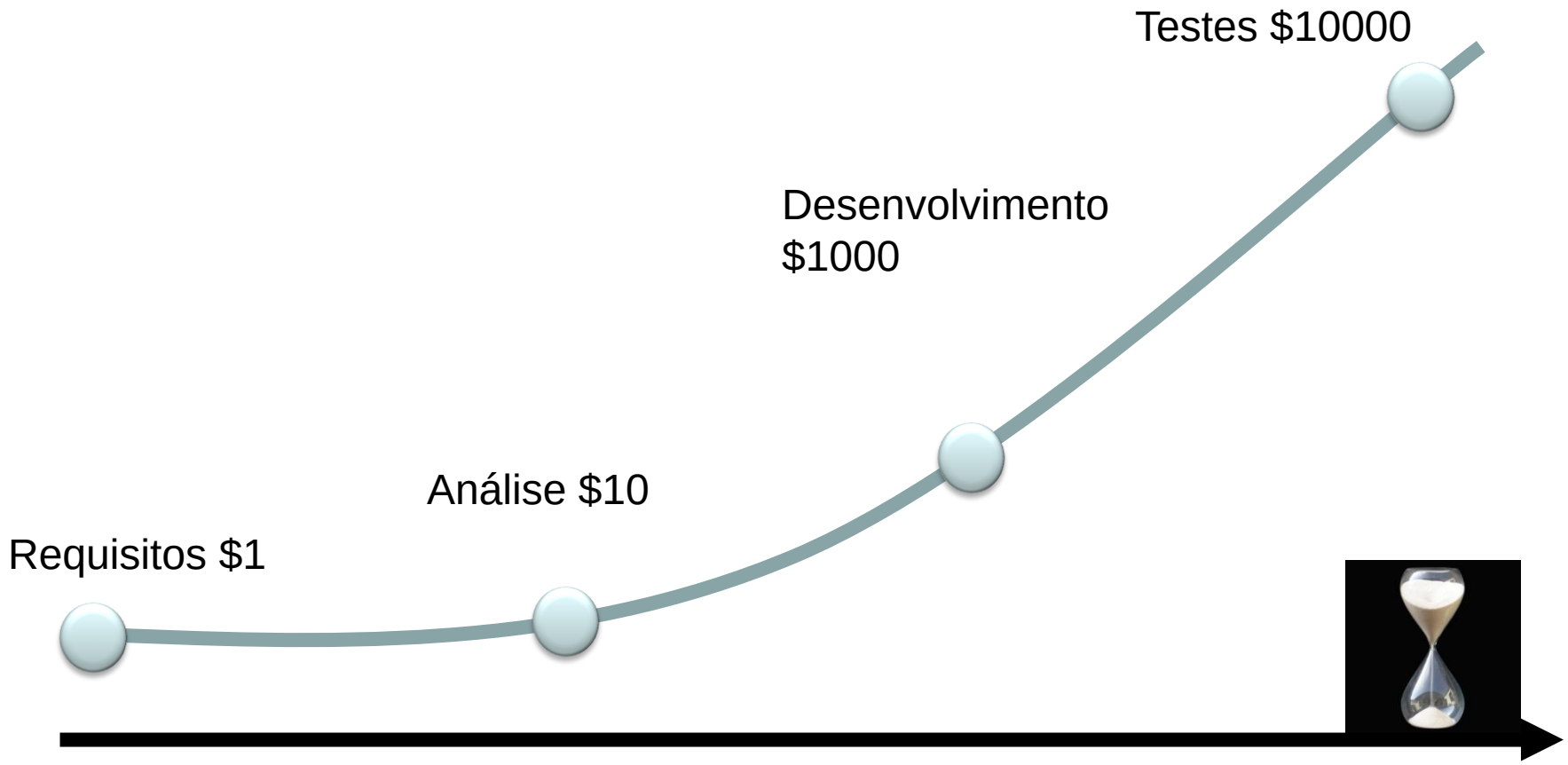
Atributo	Objetivos
Pessoas dedicadas	• Maior foco e produtividade • Equipe pequena, menos de dez pessoas
Membros de equipe multifuncional	• Desenvolver e entregar com frequência • Entregar o valor final como uma equipe independente • Integrar todas as atividades de trabalho para entregar trabalho acabado • Fornecer feedback de dentro da equipe e de outros, como o dono do produto
Agrupamento ou capacidade de gerenciar quaisquer desafios de localização	• Melhor comunicação • Dinâmica aprimorada da equipe • Compartilhamento de conhecimento • Custo reduzido de aprendizagem • Capaz de comprometer-se a trabalhar em conjunto
Equipe mista de generalistas e especialistas	• Especialistas fornecem experiência dedicada e os generalistas, a flexibilidade de quem faz o quê • A equipe traz suas habilidades especializadas e, muitas vezes, se tornam especialistas generalistas, com enfoque em uma especialidade, além de ampla experiência em múltiplas habilidades
Ambiente de trabalho estável	• Dependem uns dos outros para entregar • Abordagem aceita para o trabalho • Cálculos de custo da equipe simplificados (taxa de execução) • Preservação e expansão do capital intelectual

Modelo cascata



- Projetos reais raramente seguem o fluxo seqüencial que o modelo propõe
- Logo no início é difícil estabelecer explicitamente todos os requisitos. No começo dos projetos sempre existe uma incerteza natural
- O cliente deve ter paciência. Uma versão executável do software só fica disponível numa etapa avançada do desenvolvimento

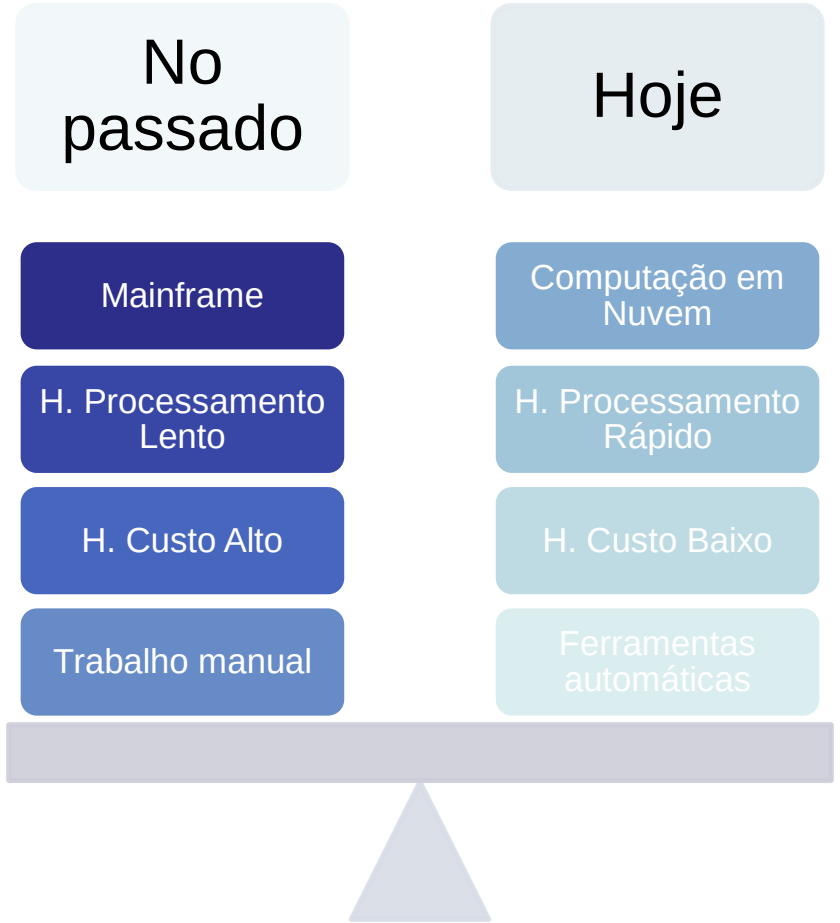
Custos das Mudanças no Modelo Cascata



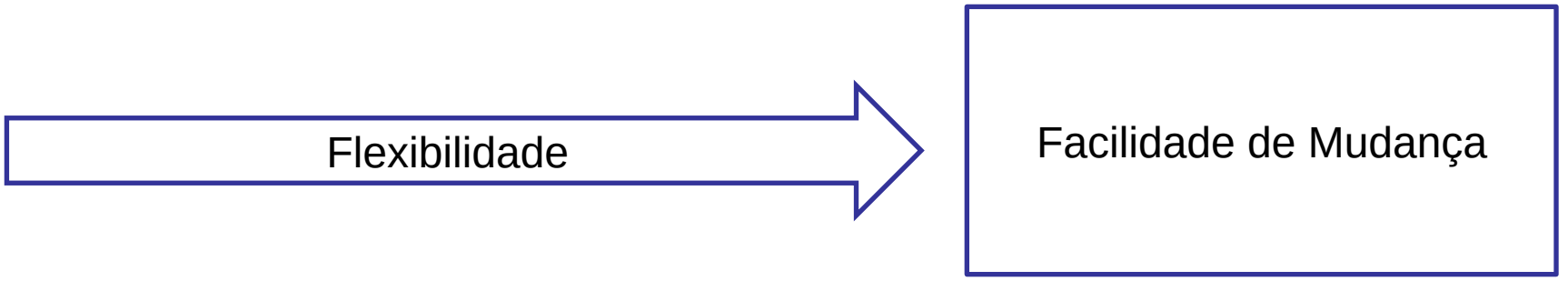
O que fazer ?

- Detectar erros e mudanças o mais cedo possível
  - Análise de requisitos
  - Modelagem (UML)
  - Protótipos

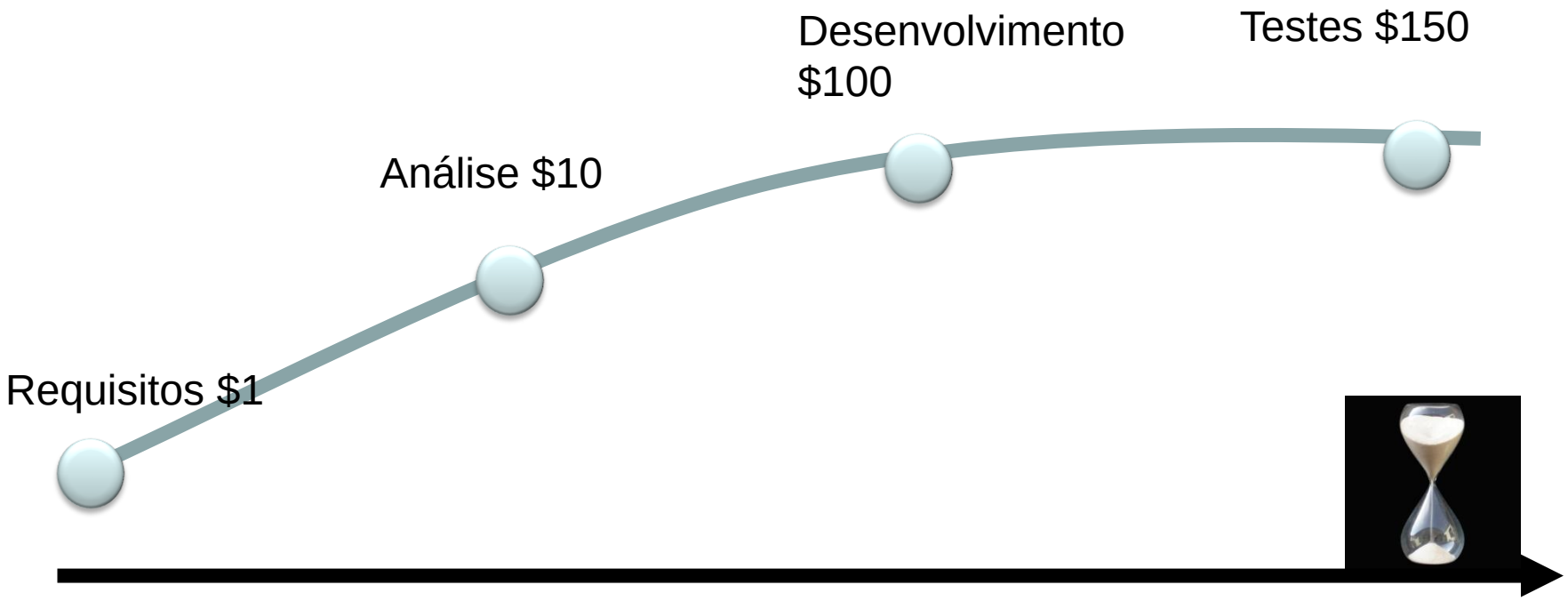
Evolução nos ultimo 30 anos



Consquencia da Evolução



Efeito da Evolução Tecnológica nos custos das Mudanças

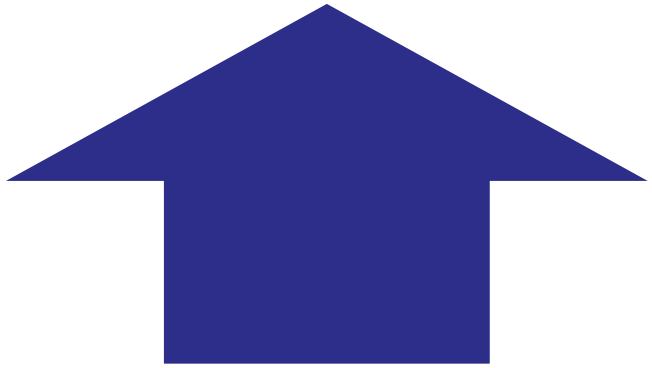


Se o custo da mudança é menos alto...

Indefinição

Mudança

Antecipar



Esperar pode agregar valor

Antecipar

Especulação

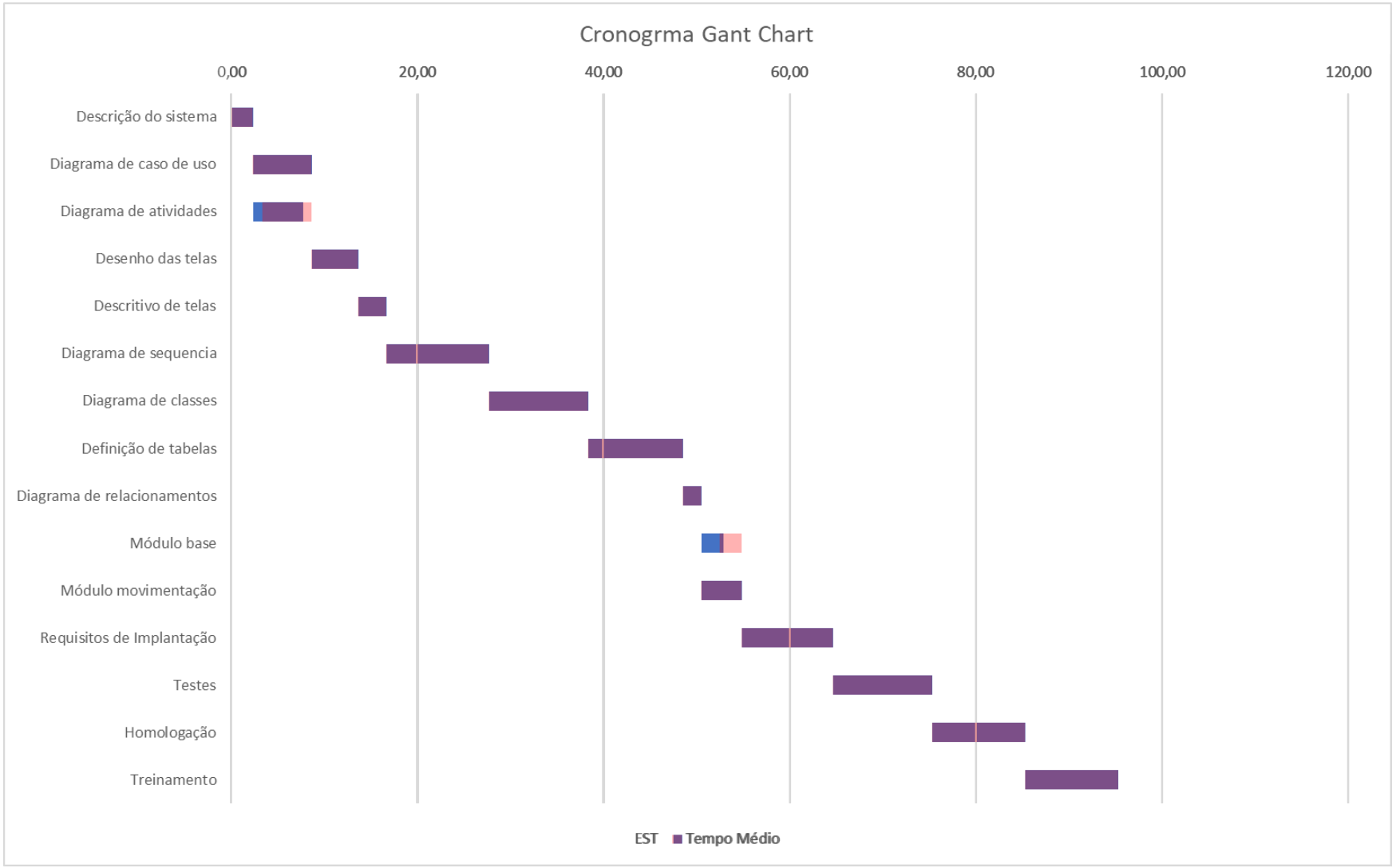
Produtividade
e bom uso das
tecnologias

Indefinição

Mudança

Pode esperar!!

Cronograma Exemplo



Manifesto Ágil (2001) Os 4 valores

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

Indivíduos e interações mais que processos e ferramentas

Software em funcionamento mais que documentação abrangente

Colaboração com o cliente mais que negociação de contratos

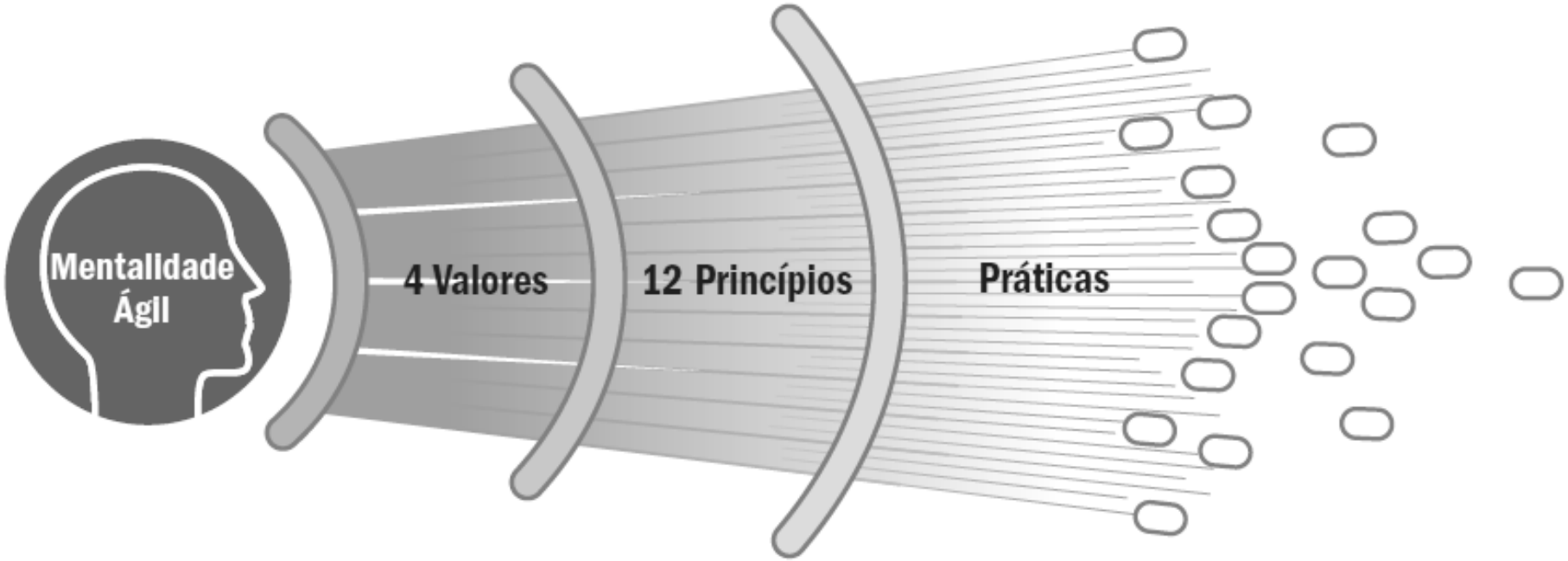
Responder às mudanças mais que seguir um plano

Ou seja, mesmo havendo valor nos itens à direita, valorizamos mais os itens à esquerda.

Manifesto Ágil - Os 12 princípios

1. A nossa maior prioridade é satisfazer o cliente por meio da entrega de valor antecipada e contínua de *software*.
2. Mudanças de requisitos são bem-vindas, ainda que tardias no desenvolvimento. Os processos ágeis tiram proveito das mudanças para a vantagem competitiva do cliente.
3. Entrega frequente de *software* funcionando, entre algumas semanas a alguns meses, de preferência no prazo mais curto.
4. Pessoas de negócio edesenvolvedores devem trabalhar juntos, diariamente, durante todo o projeto.
5. Construa projetos em torno de indivíduos motivados. Dê-lhes o ambiente e o suporte que precisam, e confie neles para concluir o trabalho.
6. O método mais eficiente e eficaz de transmitir informações para uma equipe de desenvolvimento é o comunicação face a face.
7. *Software* funcionando é a principal medida de progresso.
8. Os processos ágeis promovem o desenvolvimento sustentável. Os patrocinadores, desenvolvedores e usuários devem estar aptos a manter um ritmo constante, indefinidamente.
9. A atenção contínua à excelência técnica e ao bom design melhora a agilidade.
10. Simplicidade—a arte de maximizar o volume de trabalho não realizado—é essencial.
11. As melhores arquiteturas, requisitos e designs emergem de equipes auto-gerenciáveis.
12. Em intervalos regulares, a equipe reflete sobre como se tornar mais efetiva, depois melhora e ajusta seu comportamento de acordo com o contexto.

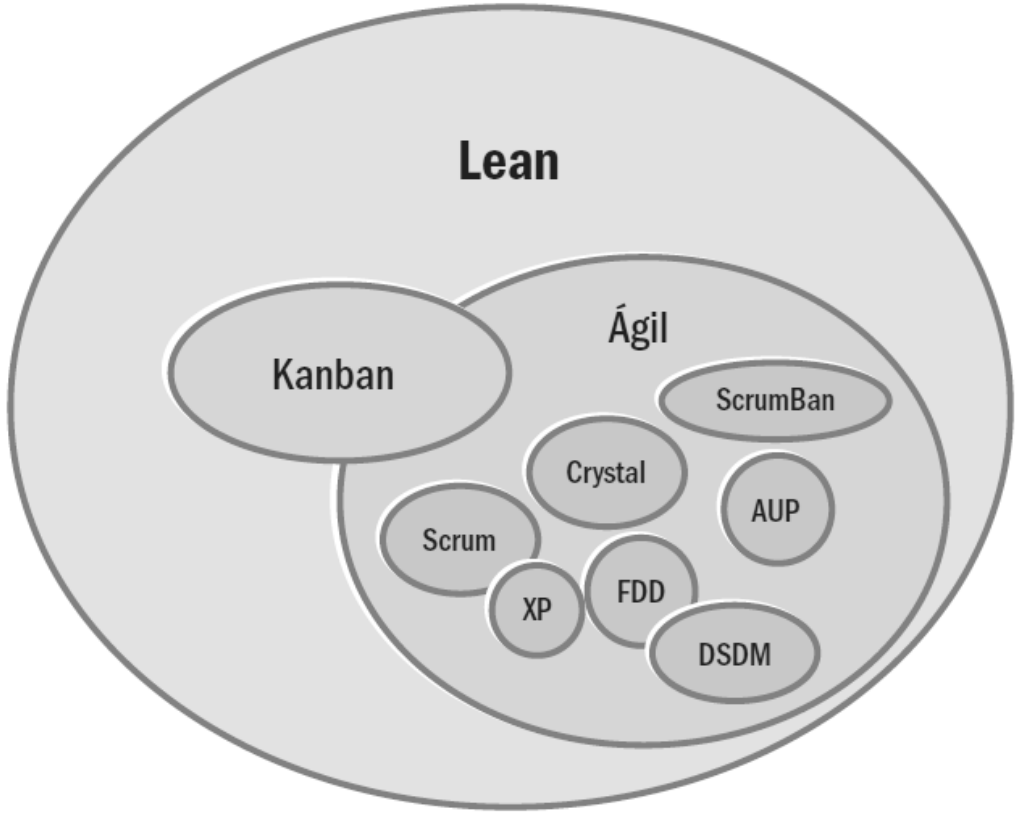
Mentalidade Ágil



Ágil é uma mentalidade definida por valores, orientada por princípios e manifestada por meio de muitas práticas diferentes. Os praticantes ágeis selecionam as práticas com base em suas necessidades.

Abordagens e Métodos Ágeis

As *abordagens ágeis* e os *métodos ágeis* são termos "guarda-chuva" que abrangem uma variedade de frameworks e métodos. A Figura 2-4 coloca a metodologia ágil em contexto e a visualiza como um termo geral, que se refere a qualquer tipo de abordagem, técnica, *framework*, método ou prática que satisfaça os valores e princípios do Manifesto Ágil. A Figura 2-4 também mostra ágil e o Método *Kanban* como subconjuntos do *lean*. Isso ocorre porque são chamados de instâncias do *lean*, que compartilham conceitos *lean* como: "foco no valor", "lotes pequenos" e "eliminação de desperdício".



Traduzindo os valores e princípios ágeis

- **Foco no código entregue**
- **Pessoas são mais importantes que processos**
- **Abordagem iterativa, retroalimentado pelo cliente**
- **Engajamento do cliente**
- **Requisitos irão mudar (e isso é bom!!)**
- **Simplicidade no código e na estrutura**

Métodos ágeis mais populares

XP (eXtreme Programming): É uma abordagem ágil para a engenharia de projetos de software. Como o próprio nome diz, é extremamente focada no desenvolvimento, e tem como principal característica a programação em par.

http://www.extremeprogramming.org/

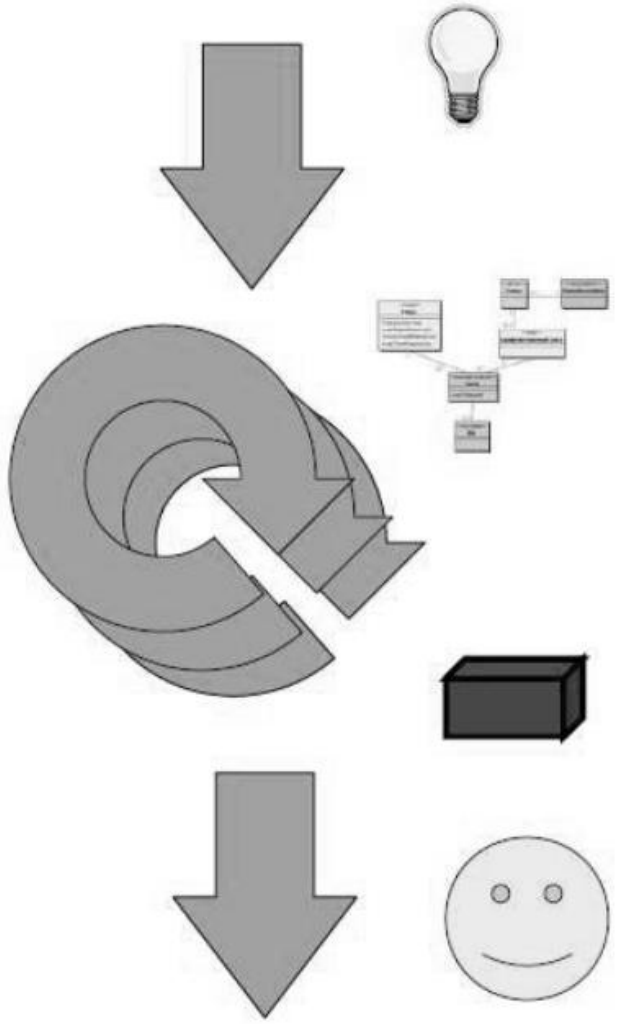
Scrum: É uma abordagem ágil para o gerenciamento de projetos. Fornece práticas que ajudam gerentes a tornar mais dinâmico e gerenciável o ambiente de desenvolvimento de software.

http://www.scrumalliance.org/

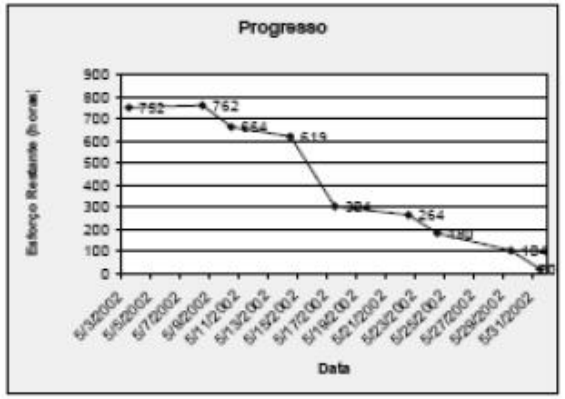
Scrum

Scrum - Fases

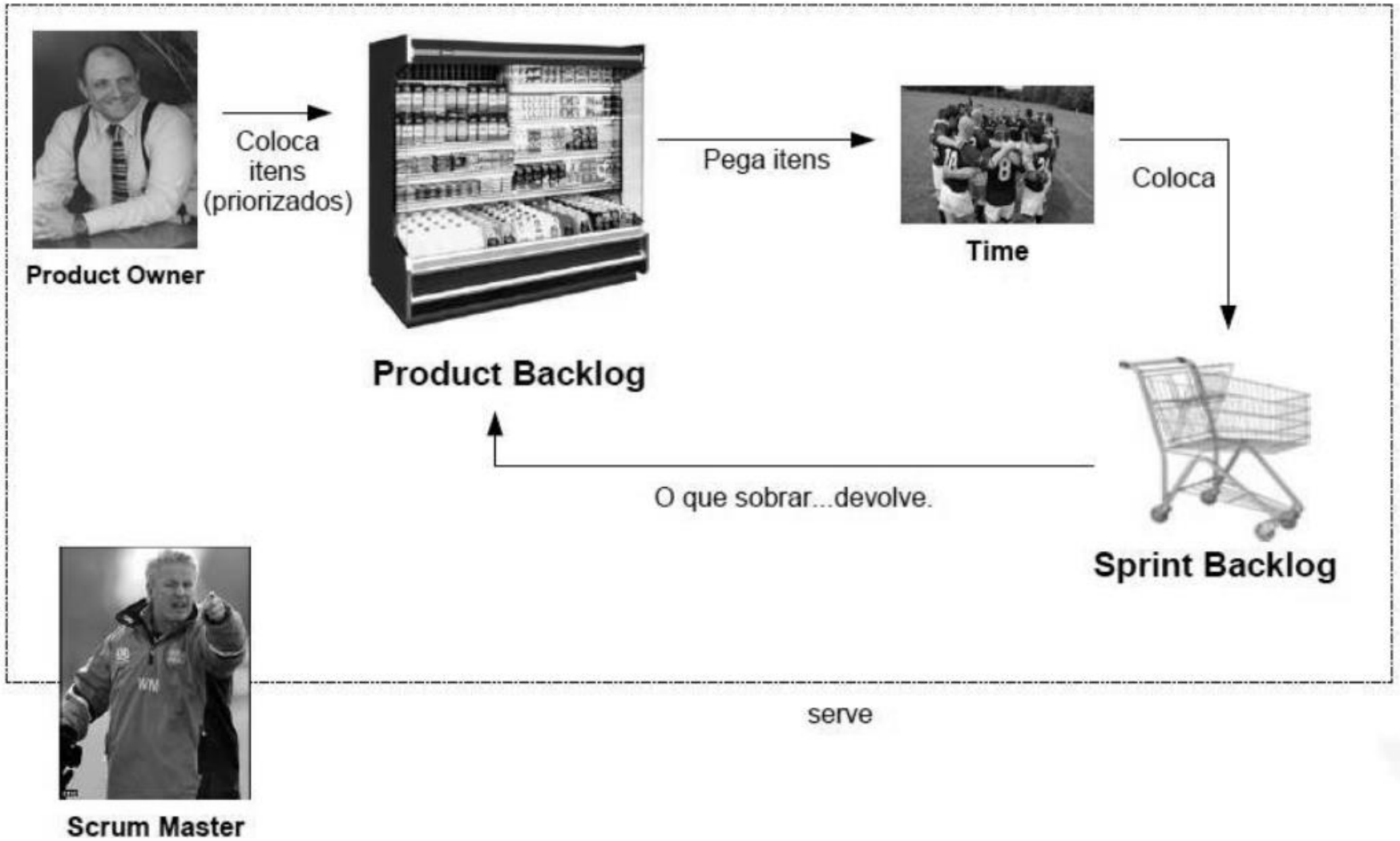
- **Pregame**
  - Planejamento
  - Desenho e alto nível da Arquitetura
  - Modelo Abrangente
- **Game**
  - Sprints (Modelagem incremental, análise, desenvolvimento, Revisões e ajustes)
- **Postgame**
  - Fechamento (Agrupamento da Documentação, Treinamento, Lições Aprendidas)



Fluxo de um sprint



Scrum Papéis



Product Backlog

- O Product Backlog é uma lista contendo todas as funcionalidades desejadas para um produto.
- O conteúdo desta lista é definido pelo Product Owner.
- O Product Backlog não precisa estar completo no início de um projeto.
- Pode-se começar com tudo aquilo que é mais óbvio em um primeiro momento.
- Com o tempo, o Product Backlog cresce e muda à medida que se aprende mais sobre o produto e seus usuários.

Product Owner

- O Product Owner é a pessoa que define os itens que compõem o Product Backlog e os prioriza nas Sprint Planning Meetings.
- O Scrum Team olha para o Product Backlog priorizado, seleciona os itens mais prioritários e se compromete a entregá-los ao final de um Sprint (iteração). Estes itens transformam-se no Sprint Backlog.
- A equipe se compromete a executar um conjunto de atividades no Sprint e o Product Owner se compromete a não trazer novos requisitos para a equipe durante o Sprint.
- Requisitos podem mudar (e mudanças são encorajadas), mas apenas fora do Sprint.
- Uma vez que a equipe comece a trabalhar em um Sprint, ela permanece concentrada no objetivo traçado para o Sprint e novos requisitos não são aceitos.

Scrum Master

- O Scrum Master procura assegurar que a equipe respeite e siga os valores e as práticas do Scrum.
- Ele também protege a equipe assegurando que ela não se comprometa excessivamente com relação àquilo que é capaz de realizar durante um Sprint.
- O Scrum Master atua como facilitador do Daily Scrum e torna-se responsável por remover quaisquer obstáculos que sejam levantados pela equipe durante essas reuniões.
- O papel de Scrum Master é tipicamente exercido por um gerente de projeto ou um líder técnico, mas em princípio pode ser qualquer pessoa da equipe.

Release Burn Down

- Em um projeto Scrum, a equipe monitora seu progresso em relação a um plano atualizando um Release Burndown Chart ao final de cada Sprint (iteração).
- O eixo horizontal de um Release Burndown Chart mostra os Sprints; o eixo vertical mostra a quantidade de trabalho que ainda precisa ser feita no início de cada Sprint.
- O trabalho que ainda resta pode ser mostrado na unidade preferencial da equipe: story points, dias ideais, team days e assim por diante.

Sprint Planning Meeting

- O Sprint Planning Meeting é uma reunião na qual estão presentes o Product Owner, o Scrum Master e todo o Scrum Team, bem como qualquer pessoa interessada que esteja representando a gerência ou o cliente.
- Durante o Sprint Planning Meeting, o Product Owner descreve as funcionalidades de maior prioridade para a equipe.
- A equipe faz perguntas durante a reunião de modo que seja capaz de quebrar as funcionalidades em tarefas técnicas, após a reunião.
- Essas tarefas irão dar origem ao Sprint Backlog.
- O Product Owner não precisa descrever todos os itens que estão no Product Backlog.
- Dependendo do tamanho do Product Backlog e da velocidade da equipe, pode ser suficiente descrever apenas os itens de maior prioridade, deixando a discussão dos itens de menor prioridade para o próximo Sprint Planning Meeting.

Sprint Planning Meeting

- Coletivamente, o Scrum Team e o Product Owner definem um objetivo para o Sprint, que é uma breve descrição daquilo que se tentará alcançar no Sprint.
- O sucesso do Sprint será avaliado mais adiante no Sprint Review Meeting em relação ao objetivo traçado para o Sprint.
- Depois do Sprint Planning Meeting, a equipe Scrum se encontra separadamente para conversar sobre o que eles escutaram e decidir quanto eles podem se comprometer a fazer no Sprint que será iniciado.
- Em alguns casos, haverá negociação com o Product Owner, mas será sempre responsabilidade da equipe determinar o quanto ela será capaz de se comprometer a fazer.

Sprint Retrospective e Review

- O Sprint Retrospective ocorre ao final de um Sprint e serve para identificar o que funcionou bem, o que pode ser melhorado e que ações serão tomadas para melhorar.
- Ao final de cada Sprint é feito um Sprint Review Meeting.
- Durante esta reunião, o Scrum Team mostra o que foi alcançado durante o Sprint. Tipicamente, isso tem o formato de um demo das novas funcionalidades.
- Os participantes do Sprint Review tipicamente incluem o Product Owner, o Scrum Team, o Scrum Master, gerência, clientes e engenheiros de outros projetos.
- Durante o Sprint Review, o projeto é avaliado em relação aos objetivos do Sprint, determinados durante o Sprint Planning Meeting.
- Idealmente, a equipe completou cada um dos itens do Product Backlog trazidos para fazer parte do Sprint, mas o importante mesmo é que a equipe atinja o objetivo geral do Sprint.

Daily Scrum

- A cada dia do Sprint a equipe faz uma reunião diária, chamada Daily Scrum.
- Ela tem como objetivo disseminar conhecimento sobre o que foi feito no dia anterior, identificar impedimentos e priorizar o trabalho a ser realizado no dia que se inicia.

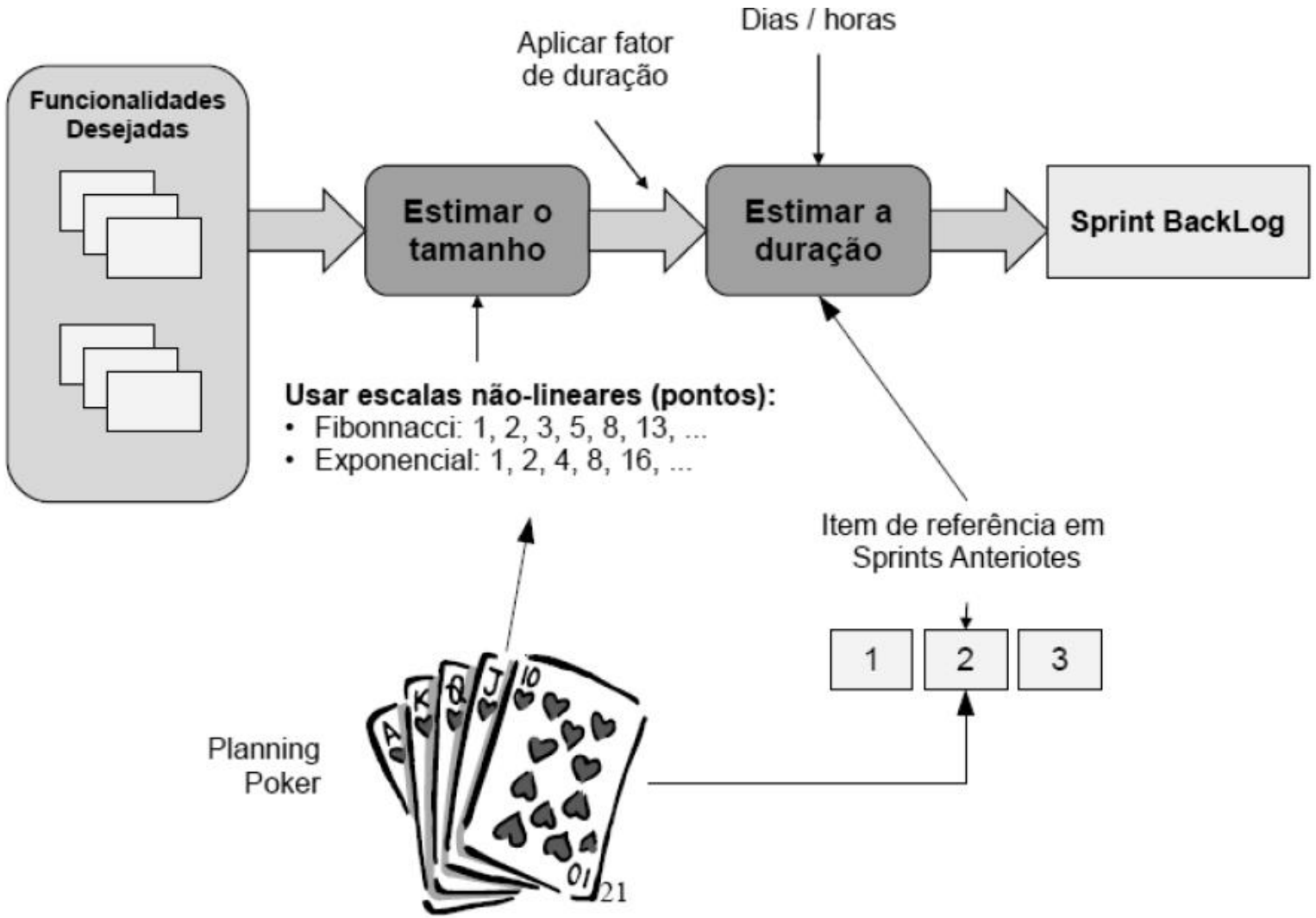
- Os Daily Scrums normalmente são realizadas no mesmo lugar, na mesma hora do dia.
- Idealmente são realizados na parte da manhã, para ajudar a estabelecer as prioridades do novo dia de trabalho.

- Todos os membros da equipe devem participar do Daily Scrum. Outras pessoas também podem estar presentes, mas só poderão escutar.
- Isso torna os Daily Scrums uma excelente forma para uma equipe disseminar informações sobre o estado do projeto.

Daily Scrum

- O Daily Scrum não deve ser usado como uma reunião para resolução de problemas.
- Questões levantadas devem ser levadas para fora da reunião e normalmente tratadas por um grupo menor de pessoas que tenham a ver diretamente com o problema ou possam contribuir para solucioná-lo.
- Durante o Daily Scrum, cada membro da equipe provê respostas para cada uma destas três perguntas:
  - O que você fez ontem?
  - O que você fará hoje?
- Há algum impedimento no seu caminho?
- Concentrando-se no que cada pessoa fez ontem e no que ela irá fazer hoje, a equipe ganha uma excelente compreensão sobre que trabalho foi feito e que trabalho ainda precisa ser feito.
- O Daily Scrum não é uma reunião de status report na qual um chefe fica coletando informações sobre quem está atrasado.
- Ao invés disso, é uma reunião na qual membros da equipe assumem compromissos perante os demais.
- Os impedimentos identificados no Daily Scrum devem ser tratados pelo Scrum Master o mais rapidamente possível.

Scrum Estimativas

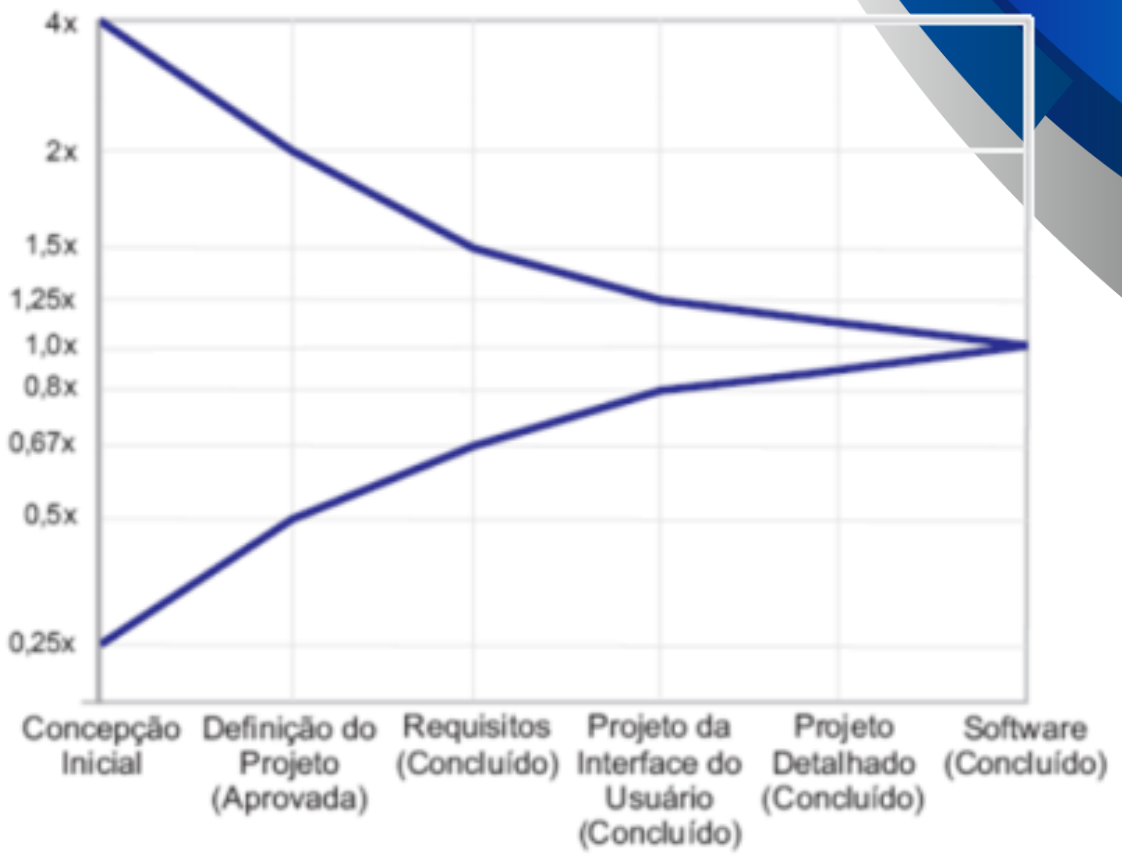


Incertezas nas Estimativas

- Qualquer que seja o tipo de atividade que você vai fazer, seja ela complexa demais ou simples demais vai existir uma variação da estimativa para o realizado.
- Em alguns momento você pode ter uma atividade muito complexa que você estima gastar 8 horas e na realidade durante a atividade você descobre alguma técnica ou procedimento e faz ela em 2 horas.
- E o contrário também pode acontecer, você estimar algo em 15 minutos e gastar 2 ou 3 horas porque surgiu algo não esperado.
- De uma vez por todas isso vai acontecer sempre!

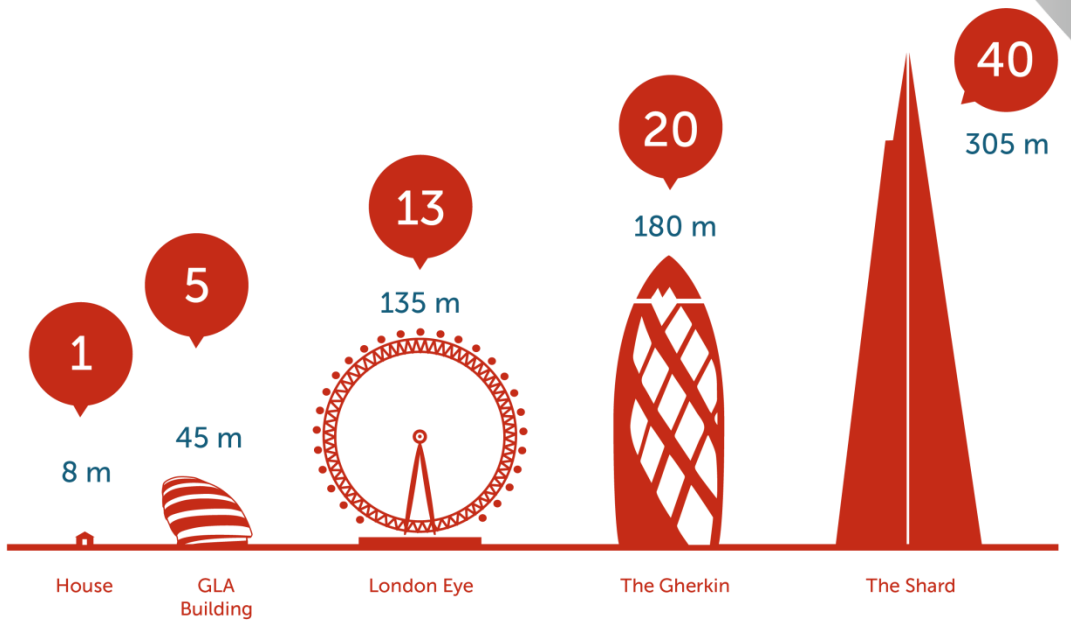
O cone das incertezas

- Em 1958 os fundadores da American Association of Cost Engineers criaram o conceito de Cone da Incerteza.
- Onde basicamente todas as estimativas realizadas levavam em conta faixas de incerteza.
- Em 1981, Barry Boehm trouxe para o desenvolvimento de software um modelo que ele mesmo batizou como Modelo Espiral,
- Esse modelo era baseado no Cone da Incerteza criado pelos engenheiros da AACE.
- O Cone da Incerteza prevê que estimativas iniciais de um projeto podem variar de 0% a 400%,
- Ou seja, a estimativa máxima pode chegar a ser 16 vezes maior que a mais baixa.
- Como você pode ver no gráfico de cone da incerteza ao lado



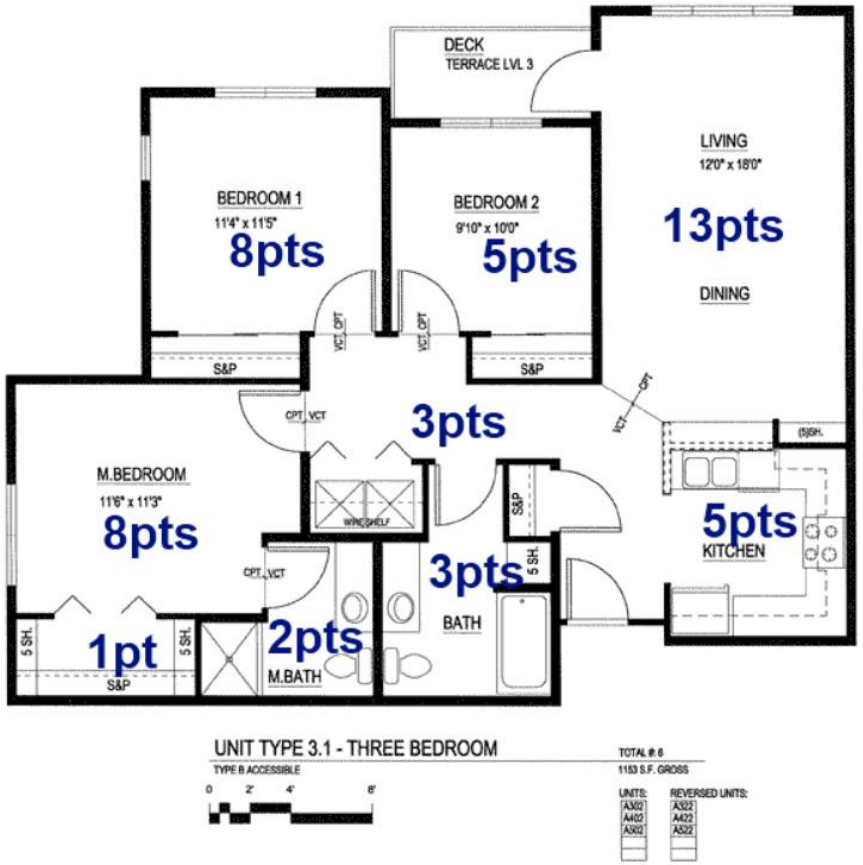
Planning Poker: Estime pelo tamanho relativo

- Em geral, somos péssimos em estimar esforço. Mas somos bons em estimar com base no tamanho relativo.
- Isso quer dizer que somos muito mais assertivos utilizando, por exemplo, P, M e G ao invés de pensar em quantas horas utilizar.
- Um bom exemplo é quando vamos comprar roupas. Antes de experimentar o que se faz?
- Comparam-se os tamanhos das peças e assim leva para experimentar.
- O interessante dessa experiência, é que ao conhecer mais a marca da roupa, na próxima compra provavelmente você irá acertar de primeira o tamanho certo.
- Esse é um exemplo de dimensionamento relativo!
- **O Planning Poker, também chamado de Scrum Poker, é uma técnica baseada em consenso e gamificada para estimar, usada principalmente para estimar o esforço ou o tamanho relativo das metas de desenvolvimento no desenvolvimento de software**



Como estimar com dimensionamento relativo – Story Points

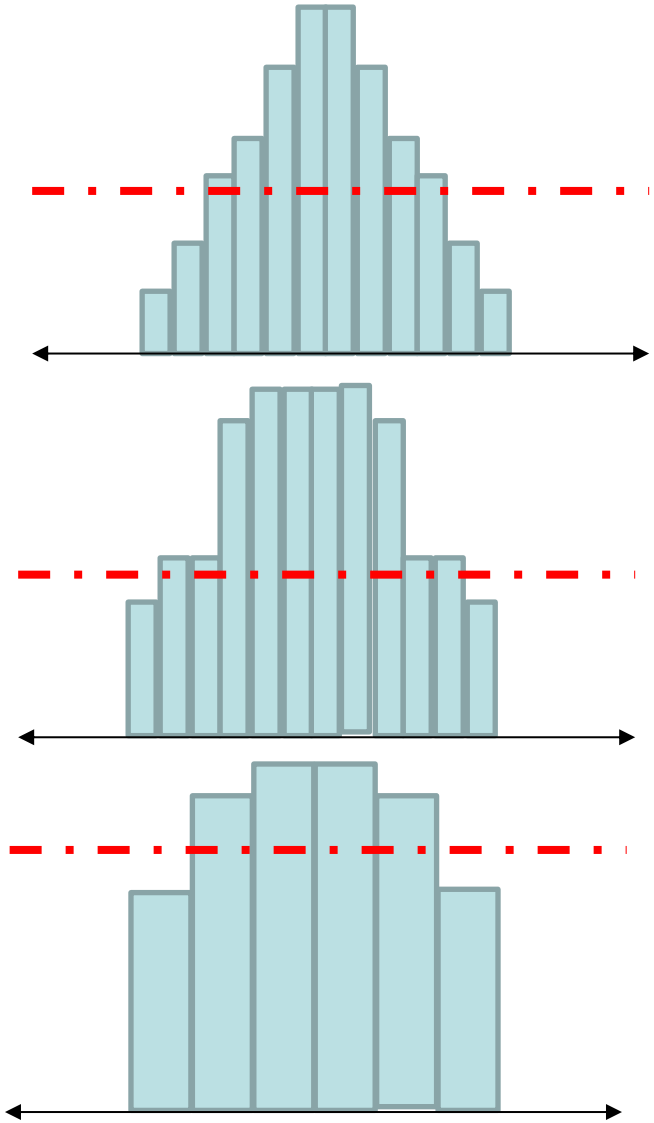
- Estimar com base na comparação e triangulação é algo bem simples, e não exige muito esforço do time.
- A única coisa que é necessária é um conhecimento prévio do que vai ser desenvolvido. Veja o exemplo ao lado,
- “um story point é uma junção da quantidade de esforço envolvido no desenvolvimento de uma feature, a complexidade desse desenvolvimento e o risco contido nele”(Cohen)
- Story point é uma medida relativa de **esforço, complexidade e risco**.
- Ou seja, quando você for estimar algo leve em consideração essas três variáveis.



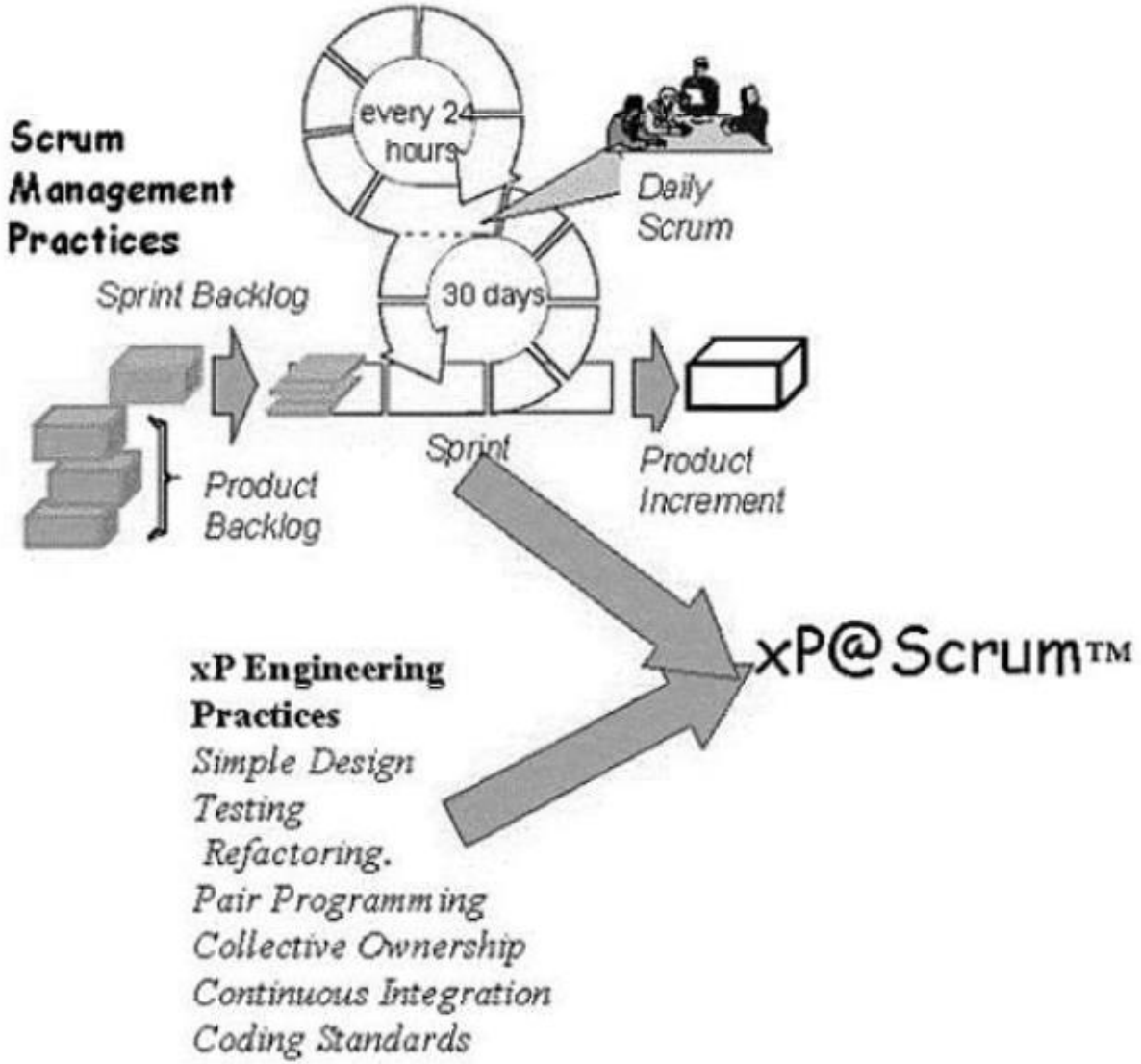
Dinâmica Planning Poker

Como ele funciona

- Alguém da equipe lê a user story em voz alta e, se possível, apresenta a mesma em um slide ou escreve no quadro branco.
- A equipe discute o critério de aceitação (definido pelo Product Owner) e tiram todas as dúvidas sobre ela.
- Cada membro, exceto o Product Owner, decide individualmente quantos Story Points e seleciona a carta do seu baralho, sem mostrar aos demais.
- Quando todos os membros já estiverem com a carta escolhida em mãos, todos viram as cartas ao mesmo tempo.
- Caso os valores sejam diferentes, cada um apresenta uma justificativa, geralmente do valor mais alto ao mais baixo.
- Depois o time vota novamente até que o grupo chegue a um acordo.



Composição XP e scrum



Scrum Ferramentas

Sprint DashBoard 01

Total BV Estimados	190

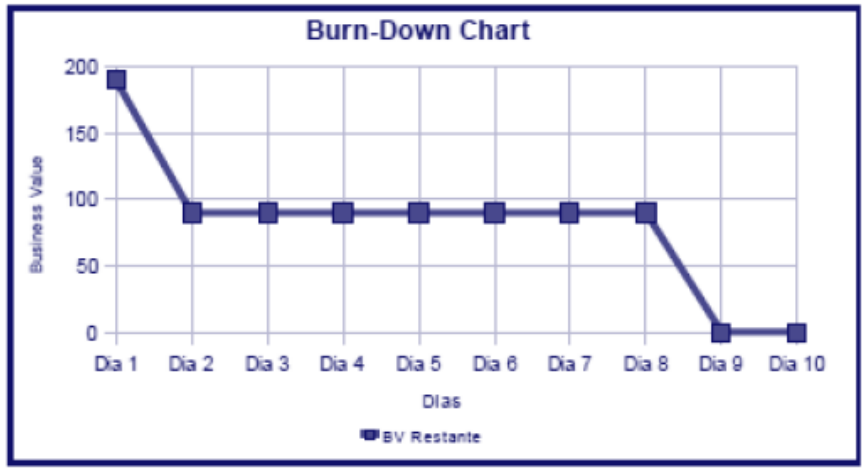
Dias	Total BV Entregues	BV Restante
Dia 1	0	190
Dia 2	100	90
Dia 3	0	90
Dia 4	0	90
Dia 5	0	90
Dia 6	0	90
Dia 7	0	90
Dia 8	0	90
Dia 9	90	0
Dia 10	0	0

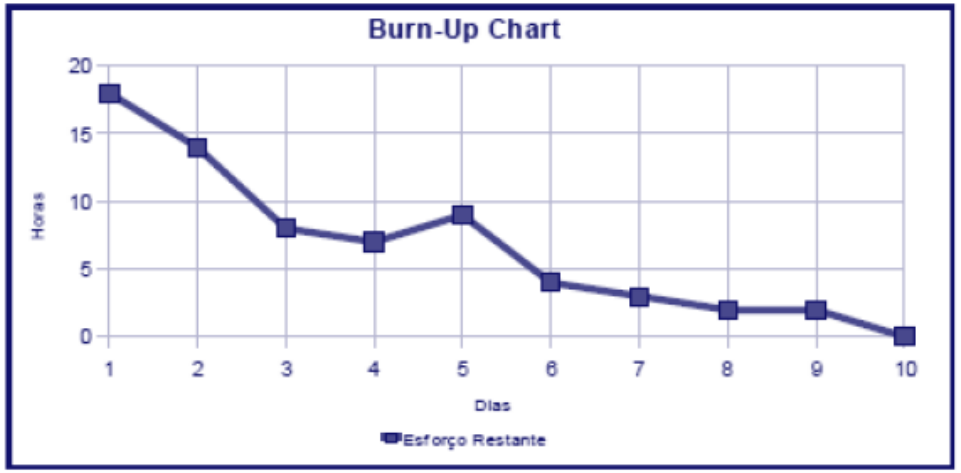
	Dias									
	1	2	3	4	5	6	7	8	9	10
Esforço Restante	18	14	8	7	9	4	3	2	2	0

Impedimentos	1

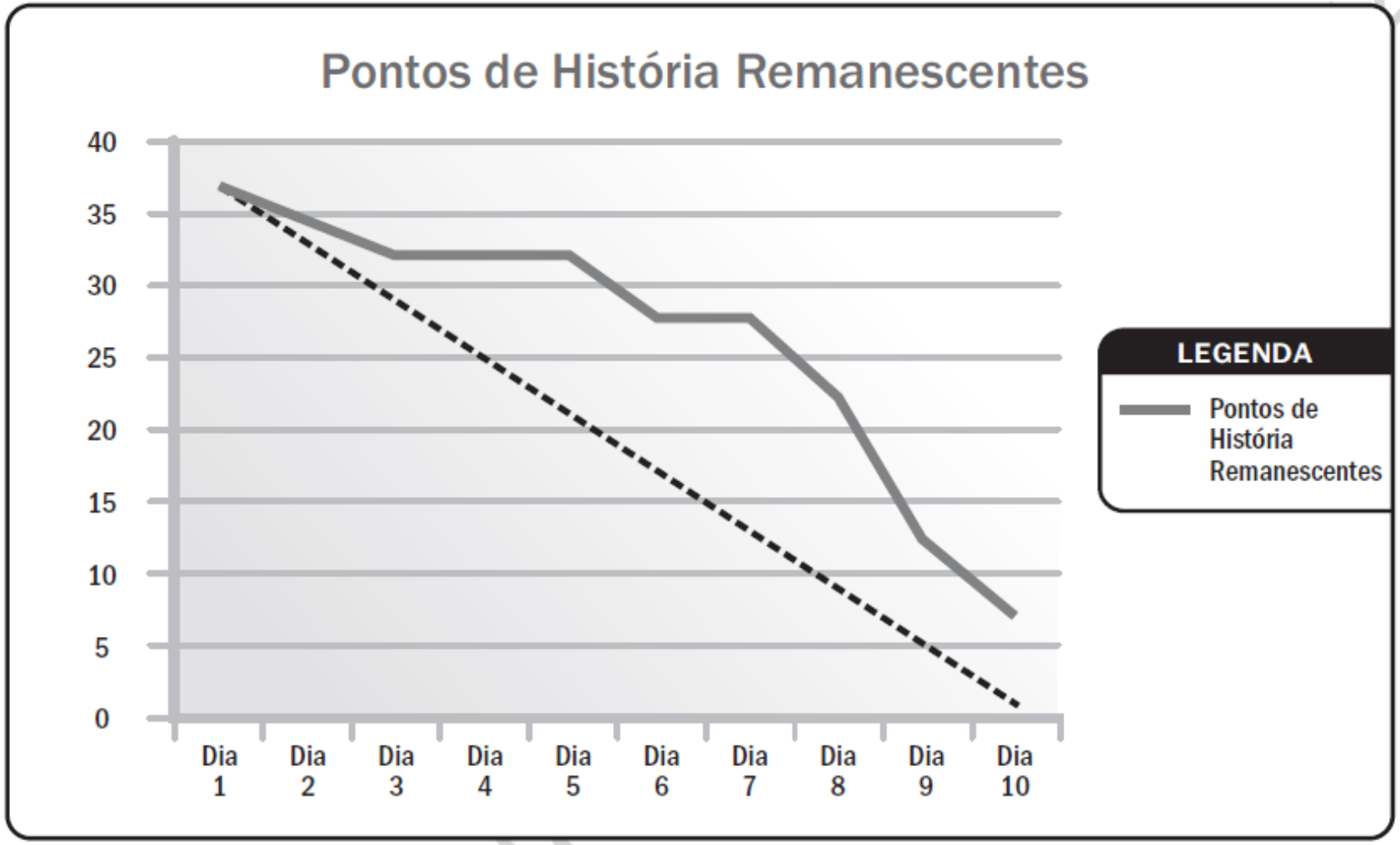
Tarefas a Fazer
2

Tarefas Finalizadas
1

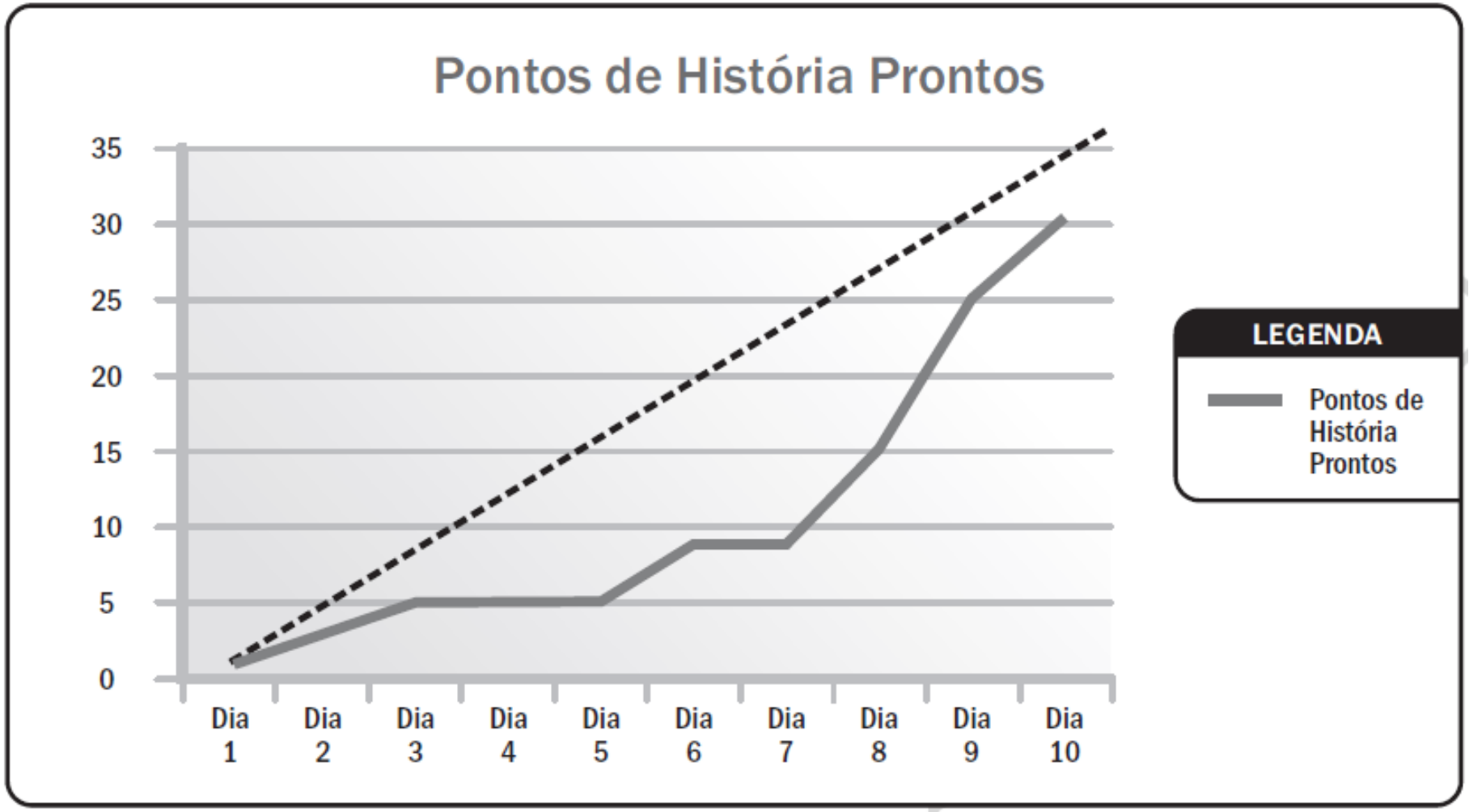




Burn Down - Pontos de história remanescentes



Burn Up – Pontos de história Prontos



Quadro Kanban

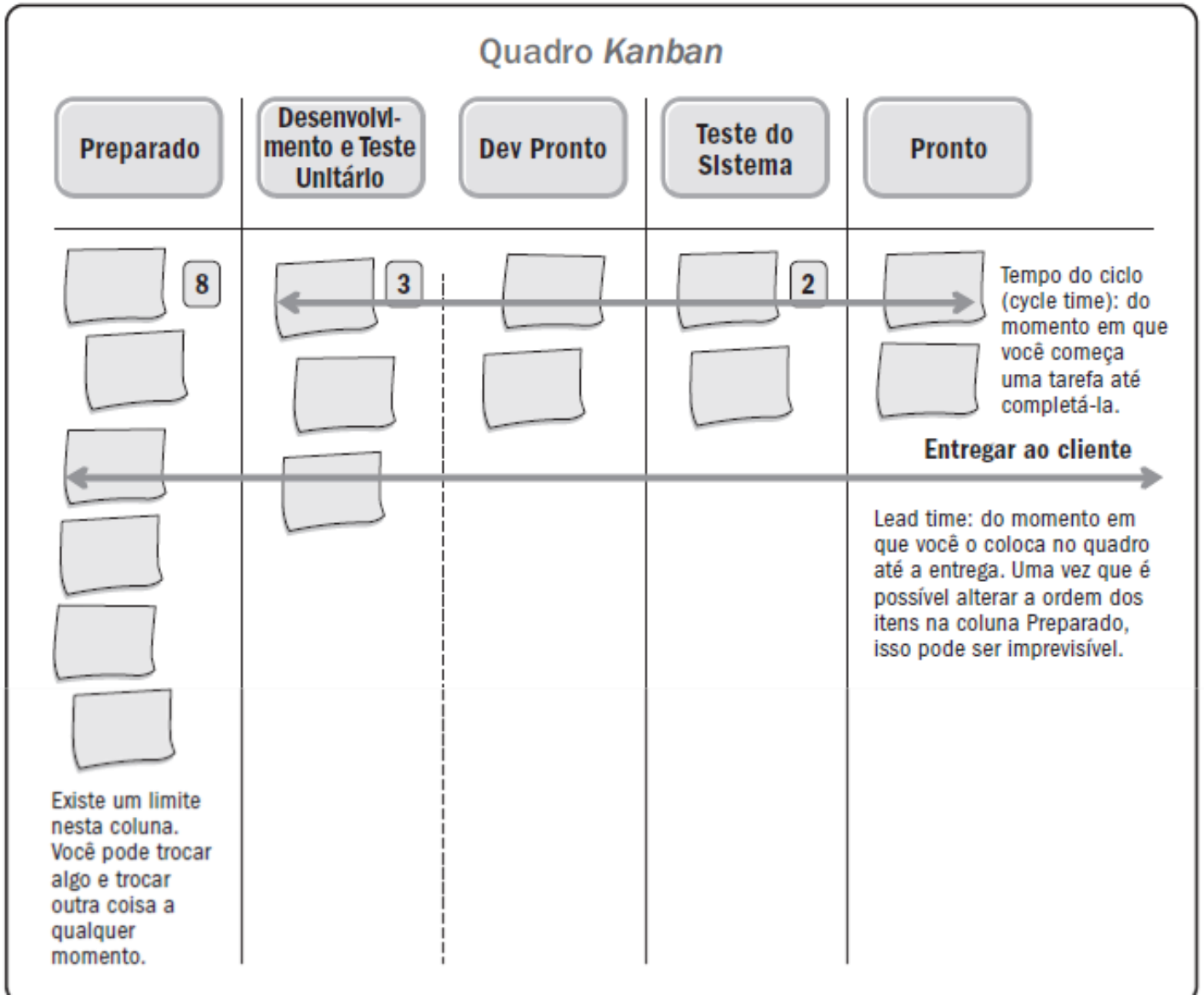


Grafico de Funcionalidades

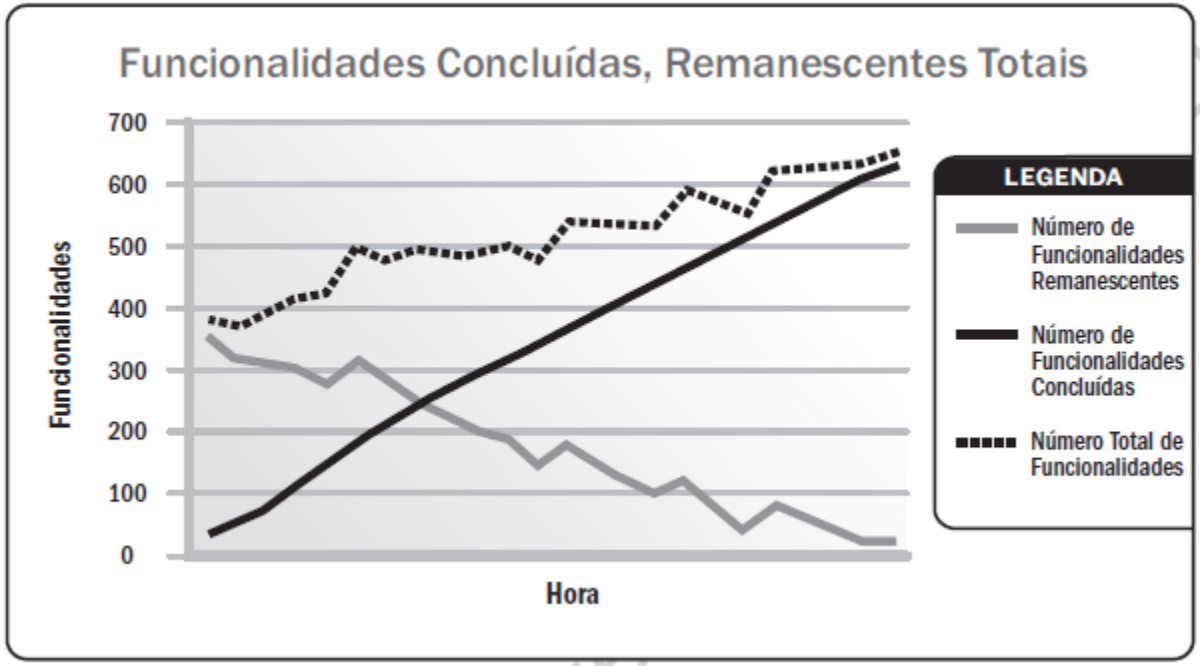


Gráfico Burn Up do Back Log

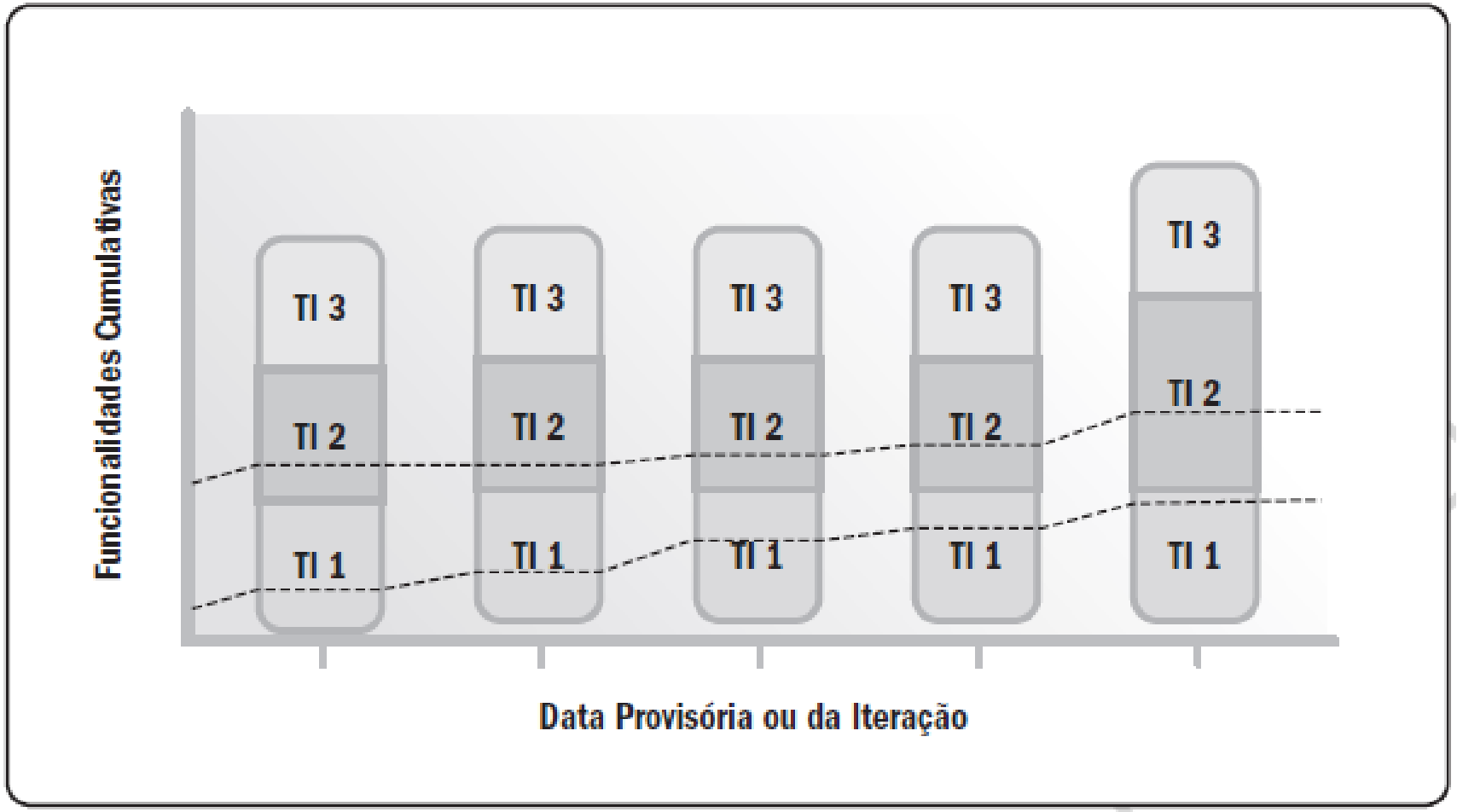
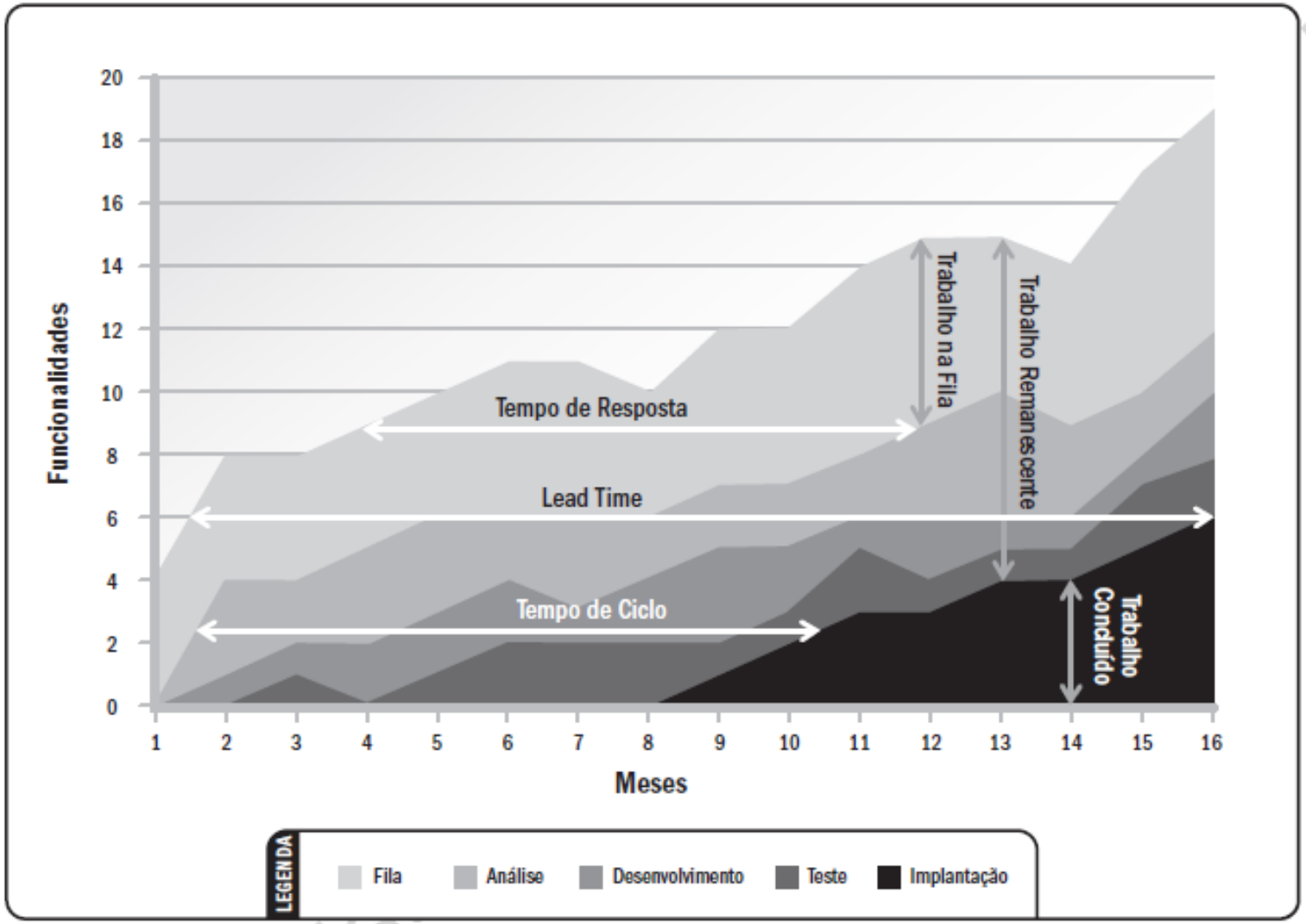


Diagrama de Fluxo Cumulativo de Funcionalidades



Extreme Programming XP

Kent Beck Estados Unidos 1999

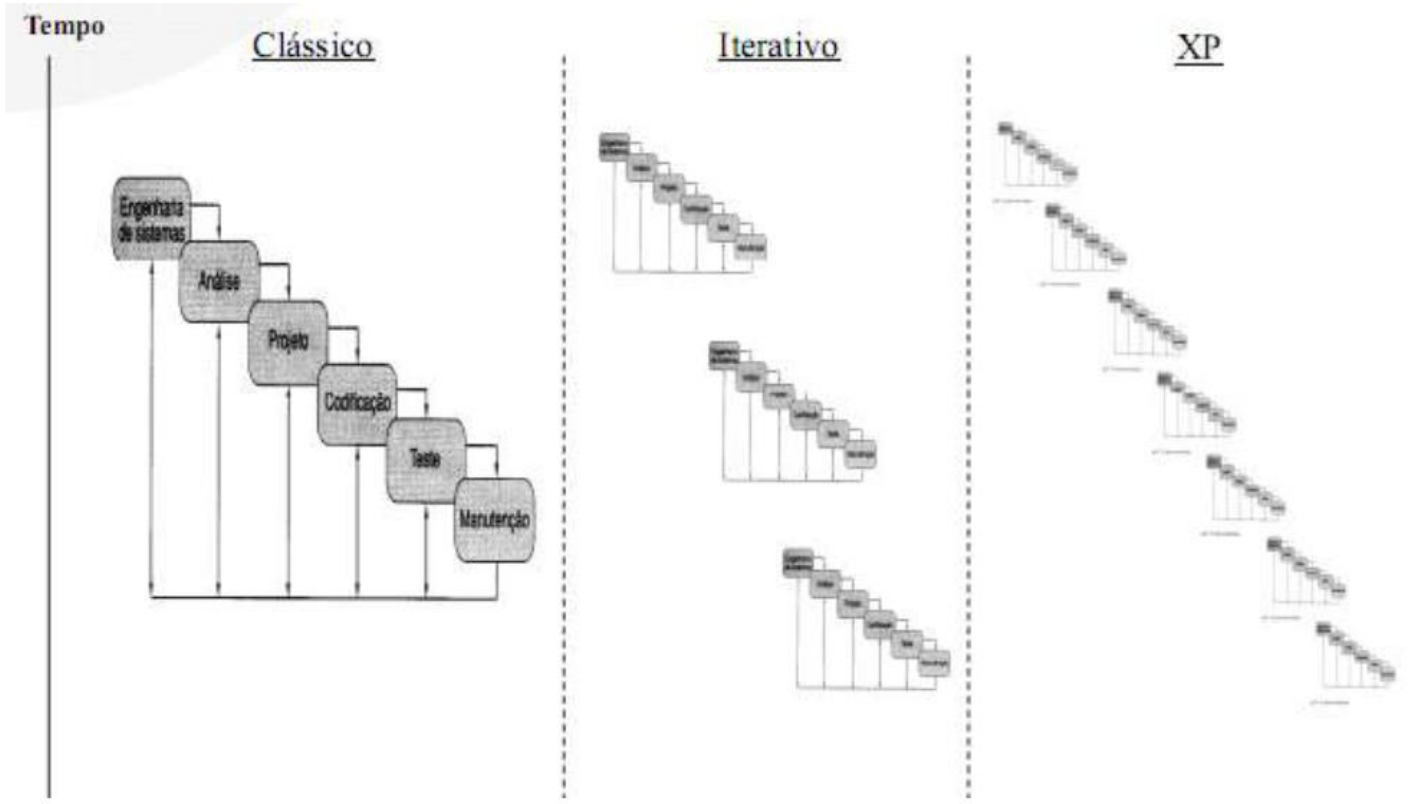
XP é leve

XP é focado no desenvolvimento de software

XP funciona em times de qualquer tamanho

XP se adapta bem a requisitos vagos e que mudam rapidamente

XP, Clássico e Iterativo

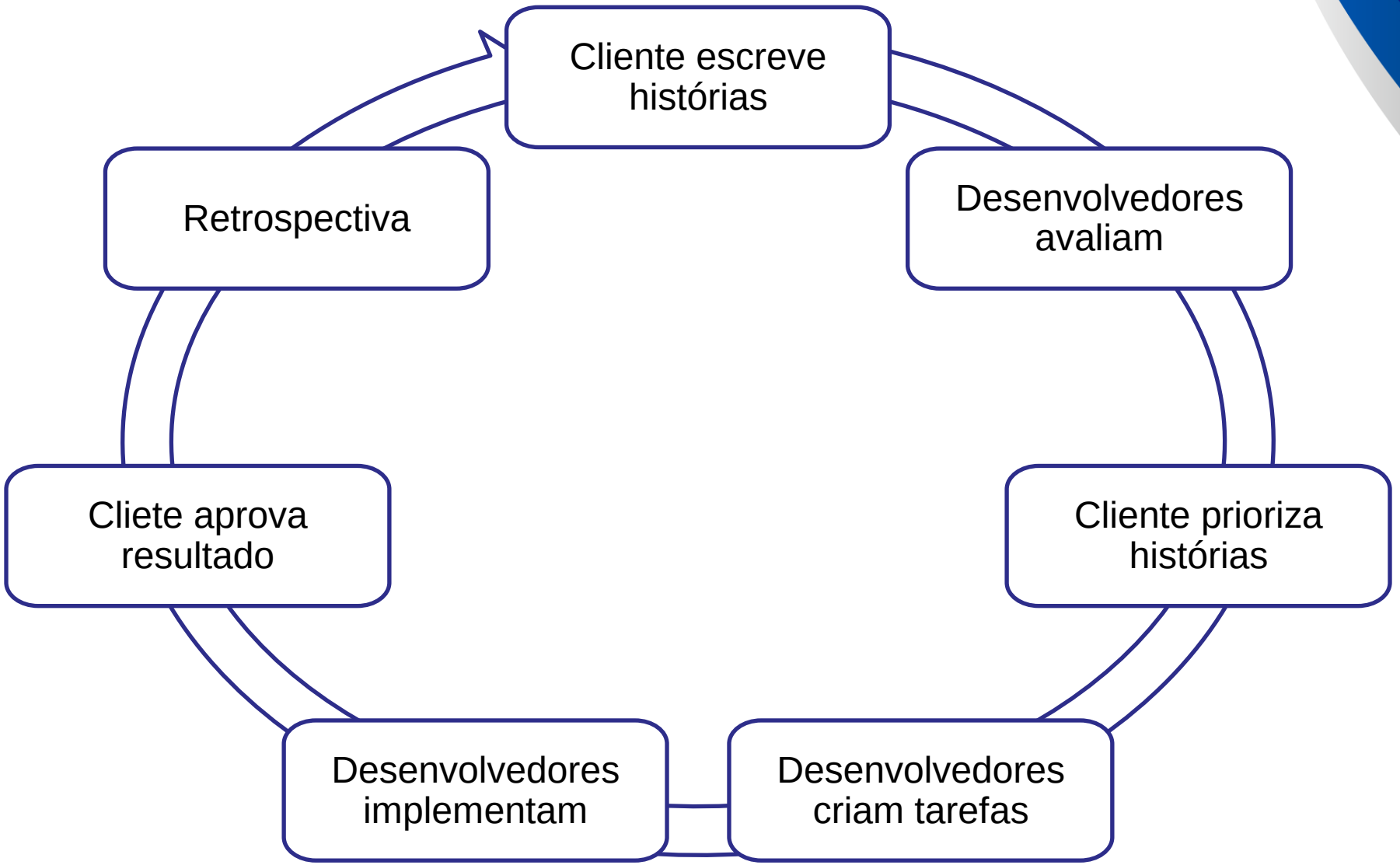


Ciclos Trimestral e Semanal

Ciclo Trimestral ⇒ Releases

Ciclo Semanal ⇒ Iterações

Ciclo Semanal



O Jogo do Planejamento

O XP utiliza o planejamento para assegurar que a equipe de desenvolvimento esteja sempre trabalhando naquilo que irá gerar mais valor para o cliente a cada dia de trabalho. Este planejamento é feito diversas vezes ao longo do projeto, para que todos tenham a oportunidade de revisar as prioridades (Beck e Fowler, 2001).

- Dividindo as Responsabilidades
- Escrevendo Estórias
- Estimando Estórias
- Planejando os Releases
- Planejando as Iterações
- Encerrando uma Iteração
- Encerrando um Release

Cliente Escreve Estórias

- Estórias exprimem o comportamento de uma funcionalidade geral
- Estórias são escritas na linguagem natural
- Formato: Who – What - Why
- Ex:
  - No papel de administrador do sistema eu quero realizar o cadastro de usuários, para armazenar informações de contato: nome, telefone e e-mail.

Análise e Planejamento

- Cada estória é analisada
- É estimado o tempo e custo para cada estória

Cliente prioriza as estórias

- Responsabilidade nas mãos do cliente
- “Aguarde e Confie”
- Conceito Chave no XP
- Limite máximo

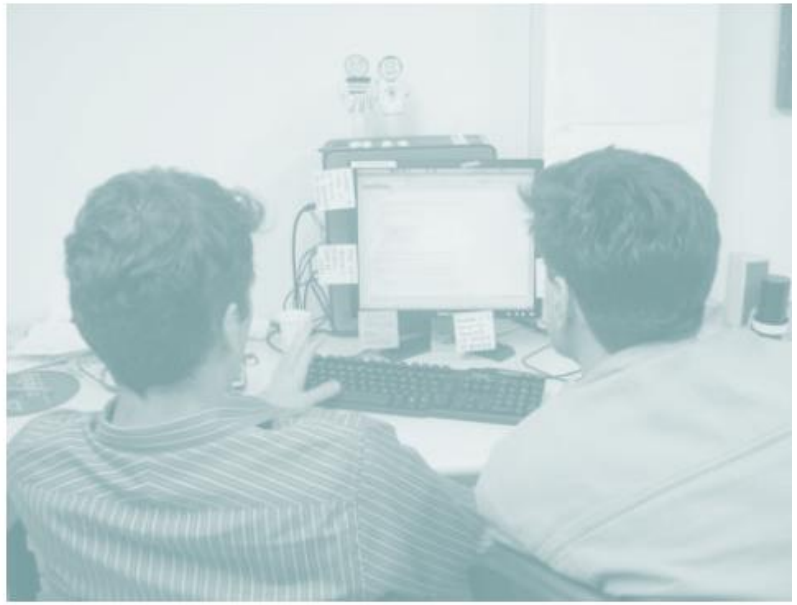
Construção das tarefas

- Ex:
  - Estória: No papel de administrador do sistema eu quero realizar o cadastro de usuários, para armazenar informações de contato: nome, telefone e e-mail.

  - Tarefas:
    - Modelagem do banco de dados
    - Criar Interface
    - Implementar cadastro

Implementação

- Programação em par
  - Todo o código
  - Um digita, outro revisa
  - Redução de bugs
  - Disseminação do conhecimento
  - Pressão do par
  - Simplicidade
  - Velocidade



Implementação

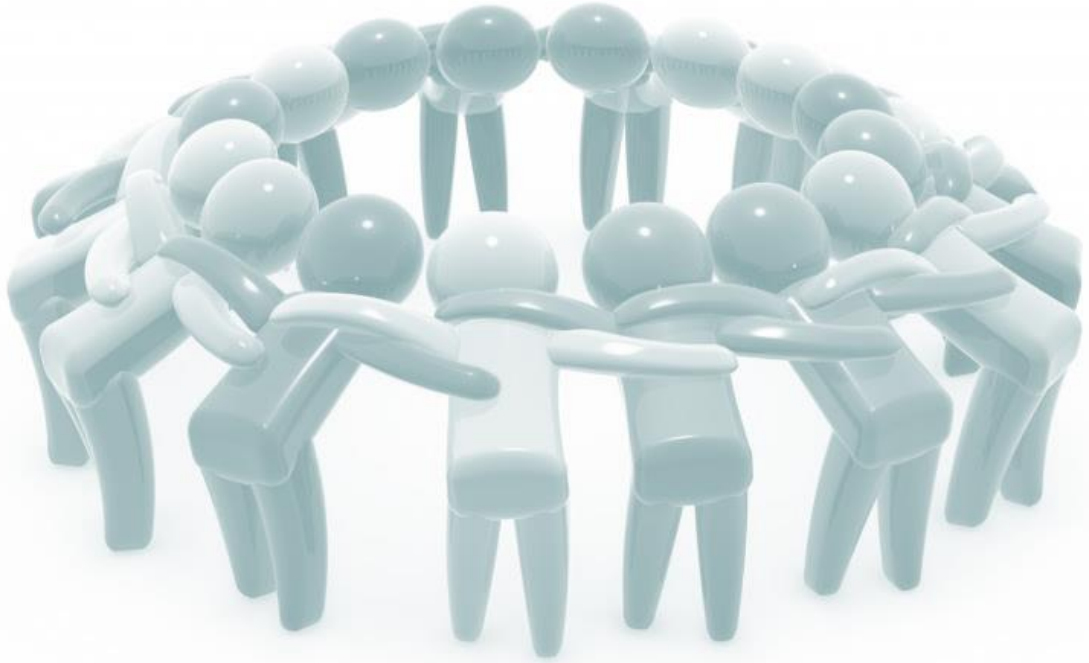
- Desenvolvimento dirigido a testes
- Propriedade coletiva do código
- Base de código unificada
- Sentar-se junto
- Refatoração

Ambiente de Trabalho

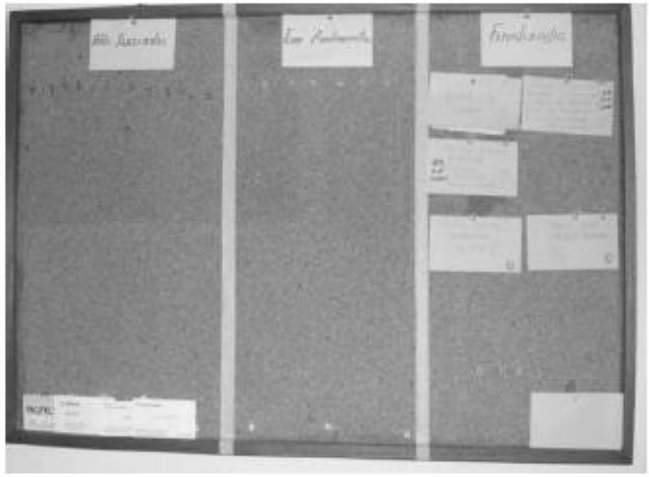
"Se você não tiver um ambiente razoável para trabalhar, seu projeto não terá sucesso" (Kent Beck)

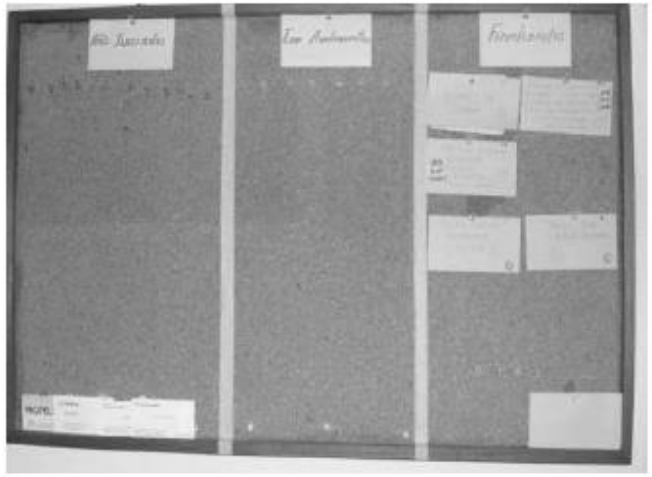
- Quadro(s) brancos
- Post-it
- Cadeiras giratórias
- Jogos
- Comida e café
- Folhas em branco
- Privacidade

Reuniões diárias



Acompanhamento





Aprovação do cliente

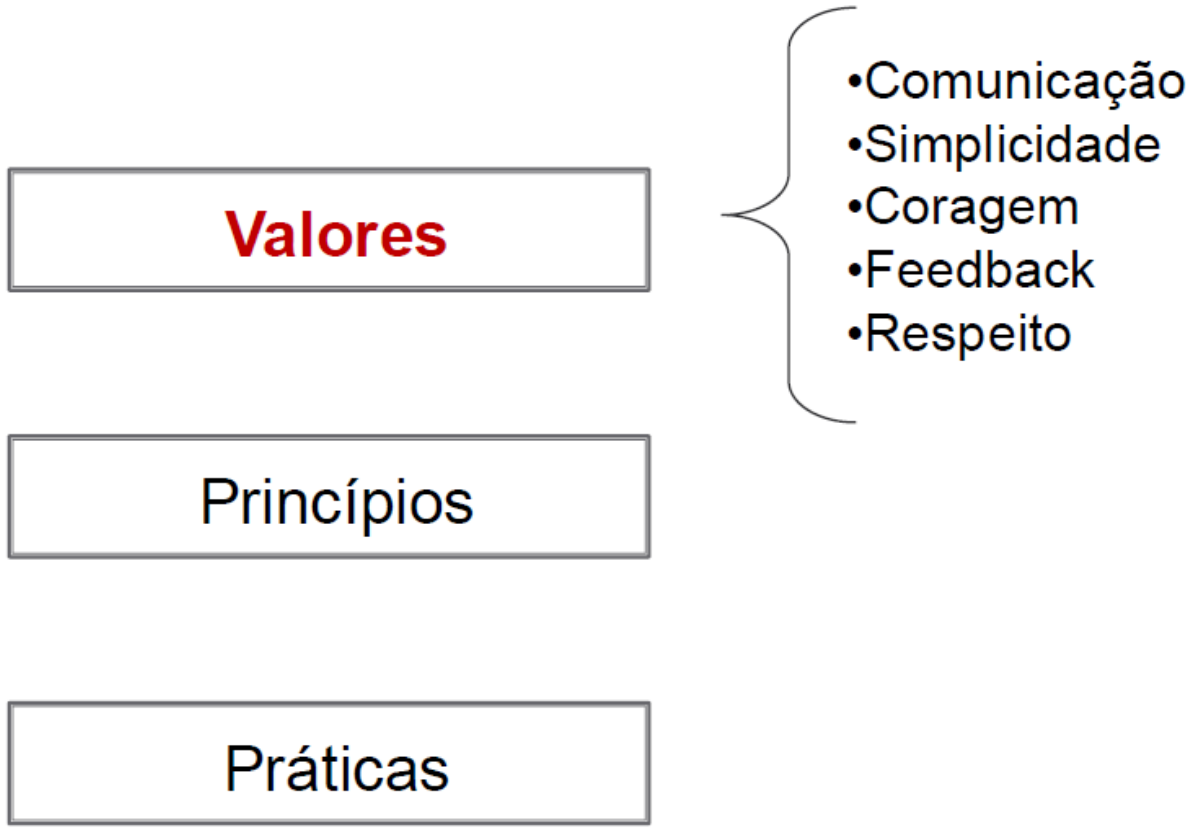
Feedback

Retrospectiva

Refactoring

- Refatoração(Refactoring): É um processo que permite a melhoria continua da programação, com o mínimo de introdução de erros e mantendo a compatibilidade com o código já existente.
- Refabricar melhora a clareza (leitura) do código, divide-o em módulos mais coesos e de maior reaproveitamento, evitando a duplicação de código-fonte;

Extreme Programming



Extreime Programming

Valores

Princípios

Práticas

- Auto-semelhança (Self-Similarity)
- Benefício Mútuo (Mutal Benefit)
- Diversidade (Diversity)
- Economia (Economics)
- Falha (Failure)
- Fluidez (Flow)
- Humanismo (Humanity)
- Melhoria (Improvement)
- Oportunidade (Opportunity)
- Passos de Bebê (Baby Steps)
- Qualidade (Quality)
- Redundância (Redundancy)
- Reflexão (Reflection)
- Responsabilidade Aceita (Accepted Respons

Extreme Programming

Valores

Princípios

Práticas

Práticas Primárias

- Ambiente Informativo (Informative workspace)
- Build de dez minutos (Ten-MinuteBuild)
- Ciclo Semanal (Weekly Cycle)
- Ciclo Trimestral (Quarterly Cycle)
- Desenvolvimento Orientado a Testes (Test-First Programming)
- Design Incremental (Incremental Desing)
- Equipe Integral (Whole Team)
- Folga (Slack)
- Histórias (Stories)
- Integração Contínua (Continuous Integration)
- Programação em Par (Pair Programming)
- Sentar-se junto (Sit together)
- Trabalho Energizado (Energized Work)

Extreme Programming

Valores

Princípios

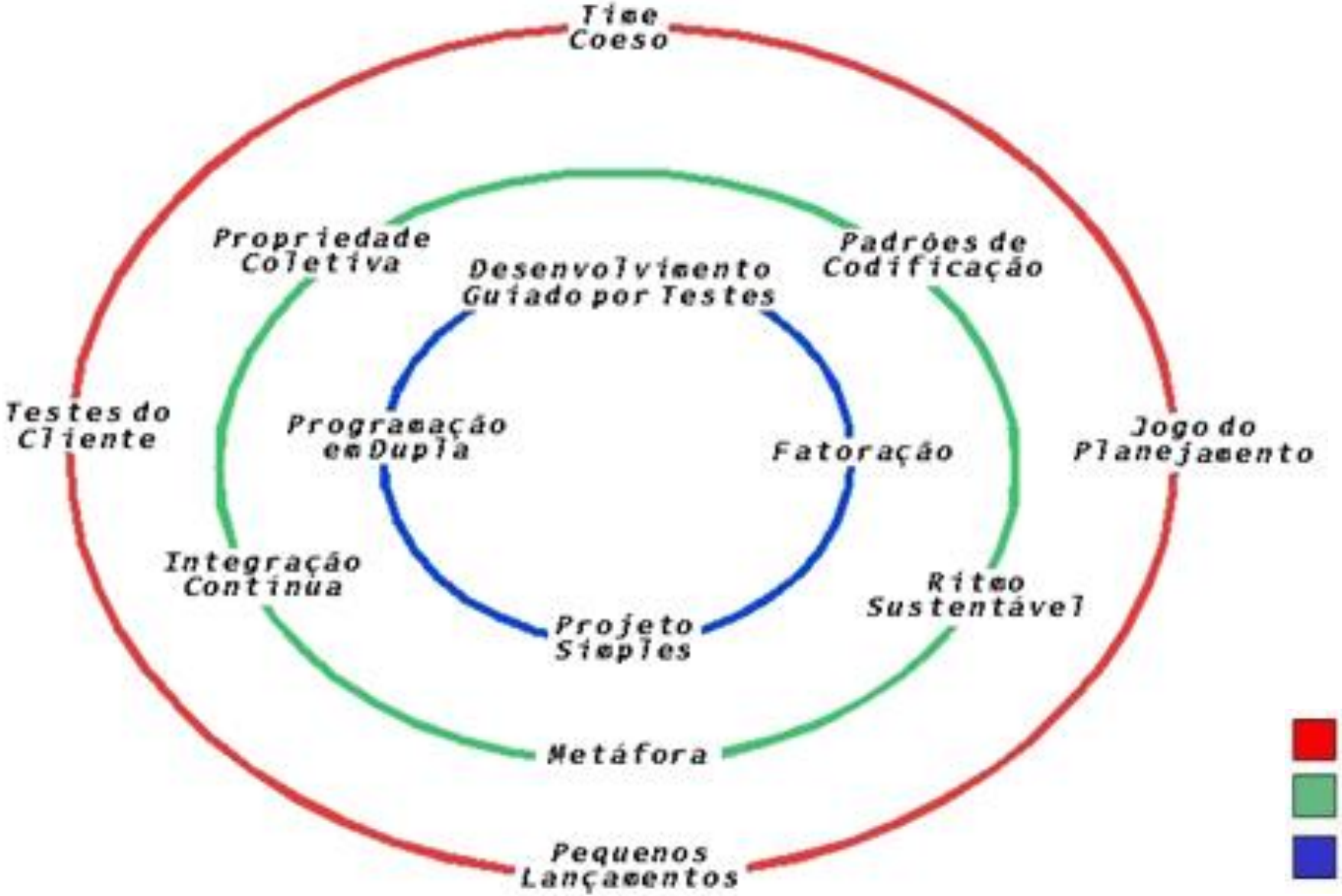
Práticas

Práticas Corolárias

- Análise da Raiz do Problema (Root-Cause Analysis)
- Base de Código Unificada (Single Code Base)
- Código Coletivo (Shared Code)
- Código e testes (Code and Tests)
- Continuidade da equipe (Team Continuity)
- Contrato de Escopo Negociável (Negotiated12 Scope Contract)
- Envolvimento do cliente Real (Real Custumer Involvement)
- Equipes que encolhem (Shrinking Teams)
- Implantação diária (Daily Deployment)
- Implantação incremental (Incremental Deployment)
- Pagar por uso (Pay-Per-Use)

Redução de riscos no XP

- Erros e falhas no cronograma
- XP pede ao cliente para priorizar cronogramas
- Confiabilidade do sistema (muitos testes são feitos)
- Má interpretação de requisitos, cliente é parte do time
- Mudança em ciclos possibilita acomodar mudanças


Time Coeso
Propriedade Coletiva
Padrões de Codificação
Desenvolvimento Guiado por Testes
Testes do Cliente
Programação em Dupla
Fatoração
Jogo do Planejamento
Integração Contínua
Ritmo Sustentável
Projeto Simples
Metáfora
Pequenos Lançamentos
Práticas organizacionais
Práticas de equipe
Práticas de pares


Equipe(Técnicos e clientes)
Propriedade Coletiva
Teste Unitário
Padrão De Código
Teste de Aceitação
Programação Em par
Refatoração
Jogo de Planejamento
Integração Contínua
Design Simples
Ritmo Sustentável
Metáforas
Pequenas Versões


IDEA!