

Histórias de Usuário

Prof. Marcelo Werneck

Disciplina: Engenharia de Requisitos

Curso: Engenharia de Software – Praça da
Liberdade – PUC Minas

Histórias - Características

- Inicialmente, unidade de funcionalidade no XP.
- Outra definição
 - Sentença curta de intenção que descreve algo que o sistema deve fazer para o usuário
- No XP histórias são escritas pelo usuário

Histórias - Características

- Scrum – Product Owner geralmente escreve histórias, apoiado pelos clientes, partes interessadas e equipe.
- Prática – qualquer membro da equipe com conhecimento de domínio pode escrever histórias.
- Product Owner aceita e prioriza histórias

Histórias - Características

- Devem ser escritas de maneira a serem compreendidas por usuários e desenvolvedores.
- Focam no valor definido pelo usuário em vez de uma decomposição hierárquica funcional.
- Podem ser escritas em cartões ou ferramentas.
- Detalhes do comportamento do sistema não aparecem na história

Histórias x Requisitos

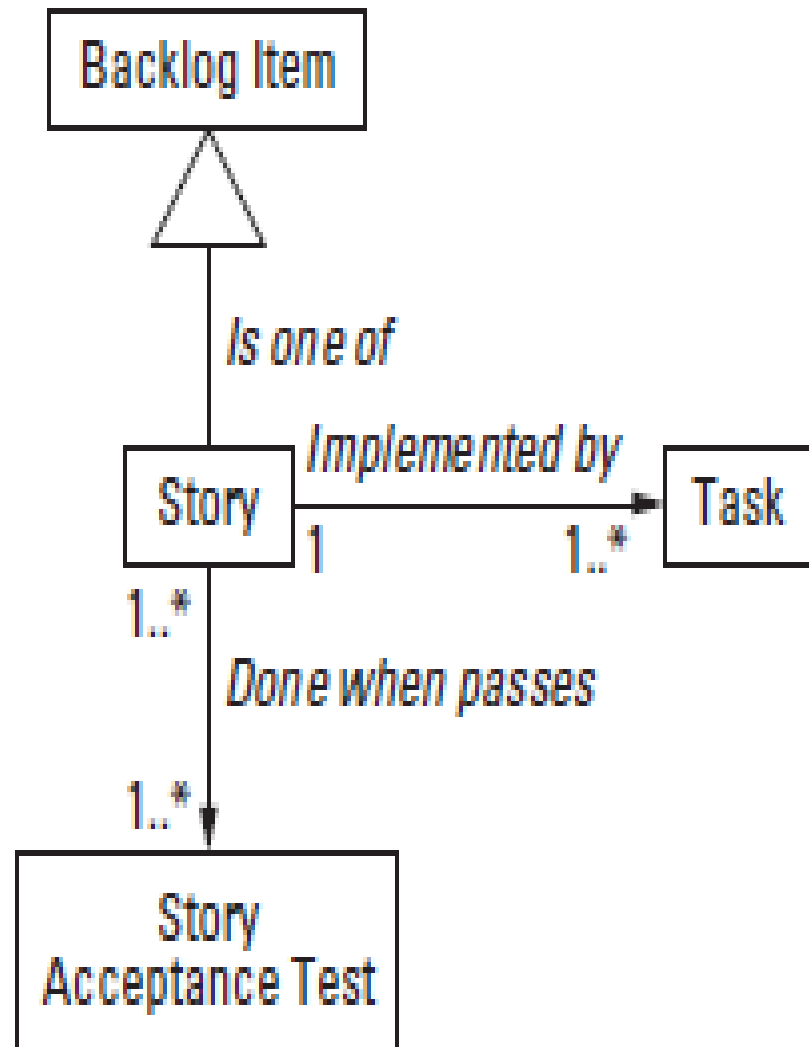
- Não são especificações detalhadas de requisitos, mas sim expressões de intenção negociáveis
- São curtas, fáceis de ler e compreensíveis a desenvolvedores, partes interessadas e usuários
- Representam pequenos incrementos de funcionalidade que podem ser desenvolvidos em dias ou semanas
- Relativamente fáceis de se estimar esforço

Histórias x Requisitos

- Não são registradas em documentos longos mas em listas organizadas
- Não são detalhadas no início do projeto mas quando necessário
- Necessitam de pouca ou nenhuma manutenção. Podem ser descartadas após a implementação.
- Servem de entrada, assim como código, para qualquer documentação posterior necessária

Histórias

- Relacionamentos



Histórias – Como escrever

- Requisitos
 - Algo que o sistema deve fornecer (necessidade do negócio ou atendimento a contrato)
- Histórias
 - Declarações curtas de intenção.
 - Algo que o sistema deve fazer para um usuário

Histórias – Como escrever

- Tem formato específico de escrita geralmente
 - Como um <papel>, eu posso/devo <ação> para que <valor para o negócio>
- Foco:
 - Papel do usuário
 - Valor para o negócio



Histórias – Como escrever

- Papel
 - Permite uma segmentação da funcionalidade do produto
 - Identifica necessidades baseadas em papéis.
- Atividade
 - Representa “requisito do sistema”
- Valor para o negócio
 - Comunica porque a atividade é necessária
 - Atividades alternativas podem ser discutidas

Histórias – Como escrever

- Foco é no valor para o negócio e na resolução de problemas reais para pessoas reais!!!

Histórias - Exemplos

1. User stories

1.1 Most popular blu-ray discs sold

As a **customer** I want to **see the most popular blu-ray discs sold** so that I **can order one or many of them**

1.1.1 Sort by price

As a **customer** I want to **sort the most sold blu-ray discs by price** so that I **can see the less expensive ones first**

1.1.2 Sort by popularity

As a **customer** I want to **sort the most sold blu-ray discs by popularity** so that I **can see the most popular ones first**

Histórias – Mais exemplos

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

As a librarian, I
want to be able
to search for books
by publication year.

Processo completo de uso

- Relacionada à abordagem de entrega de valor ao cliente
 - Defina uma história de valor para o usuário
 - Implemente e teste em uma iteração curta
 - Demostre ou entregue para o usuário

Representação das histórias

- Cartão
 - Representa duas ou três frases usadas para descrever a intenção da história
 - Detalhe ainda será determinado
 - Promessa de conversa
- Conversa
 - Discussão entre a equipe, cliente e product owner para determinar comportamento detalhado.
- Confirmação
 - Representa o teste de aceitação
 - Representa condições de satisfação

Detalhamento das histórias

- Detalhes são alcançados principalmente pela conversação.
- Se mais detalhes são necessários, podem ser providos por anexos
 - Mock-up, planilha ou algoritmos
- Detalhes são coletados na medida em que se tornam necessários (*just in time*)

Boas histórias

- Como escrever boas histórias?
- Algumas características são necessárias
 - Independentes
 - Negociáveis
 - Valor
 - Estimáveis
 - São de tamanho pequeno (Small)
 - Testáveis

Características de boas histórias

- Independentes
 - História pode ser desenvolvida, testada e até entregue isoladamente.
 - A conclusão de uma história deixa o produto em um estado em que ele pode ser entregue
- Negociáveis
 - Não são contrato de implementação de algo
 - Oportunidade de discussão dos requisitos
 - Processo de colaboração
 - Deve haver possibilidade de trade-offs entre funcionalidade e datas de entrega
 - Deve haver flexibilidade para atender objetivos

Características de boas histórias

- De Valor
 - Essência dos métodos ágeis
 - Backlogs são priorizados por valor
 - Mudança de uma abordagem de decomposição hierárquica funcional para abordagem vertical
 - Ex: fatiar o bolo



Características de boas histórias

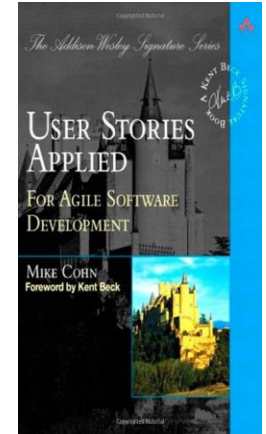
- Estimáveis
 - Equipe deve ser capaz de prover uma estimativa de sua complexidade e quantidade de trabalho necessário para terminar a história
 - Ao menos deve ser completada em uma iteração
 - Se equipe não consegue estimar, história deve ser quebrada em histórias menores
 - Pode haver uso de “spikes” funcionais ou técnicos para aumentar assertividade e tomar decisão
 - Conversa durante o processo de estimativa é importante para levantar requisitos implícitos

Características de boas histórias

- São de tamanho pequeno (Small)
 - Completadas em uma iteração
 - Histórias pequenas provêm maior agilidade
- Testáveis
 - Abordagem de se escrever o teste antes
 - TDD (Test Driven Development)
 - Se equipe sabe como testar uma história, eles sabem como codificá-la.
 - Palavras como “rapidamente”, “gerenciar”, “limpo” são fáceis de escrever mas ambíguas. Devem ser evitadas

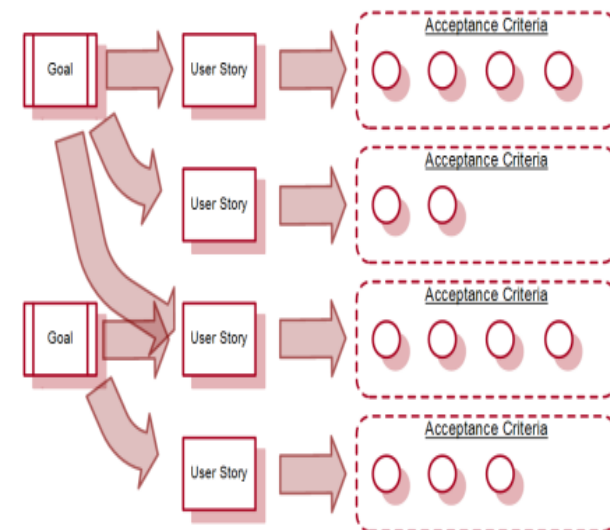
Diretrizes para boas histórias

- ✓ O que já foi visto sobre histórias de usuário:
 - ✓ Como escrever histórias
 - ✓ Como identificar histórias
 - ✓ Papéis relacionados
 - ✓ Testes de aceitação
- ✓ É possível traçar algumas diretrizes adicionais para a escrita de boas histórias.
- ✓ Os exemplos apresentados aqui são do livro “*User stories applied*”.



Começar com objetivos do sistema

- ✓ Em projetos grandes com muitos usuários
 - ✓ Difícil saber por onde começar
- ✓ Sugestão é considerar cada papel e começar pelos seus objetivos de maior prioridade.
- ✓ Objetivos são histórias de alto-nível.
 - ✓ Irão gerar histórias adicionais.



Fatiar o bolo

- ✓ Há várias maneiras de se quebrar uma história em pedaços menores.
- ✓ Tendência se realizar quebra considerando aspectos técnicos.
- ✓ Melhor abordagem: Fatiar o bolo.
 - ✓ Cada história deve conter algum nível completo de funcionalidade.
 - ✓ Uma maneira também de se testar camadas da arquitetura.
 - ✓ Aplicação pode até ser entregue com funcionalidade parcial mas que inclui todas as camadas.



Ciclos trimestrais

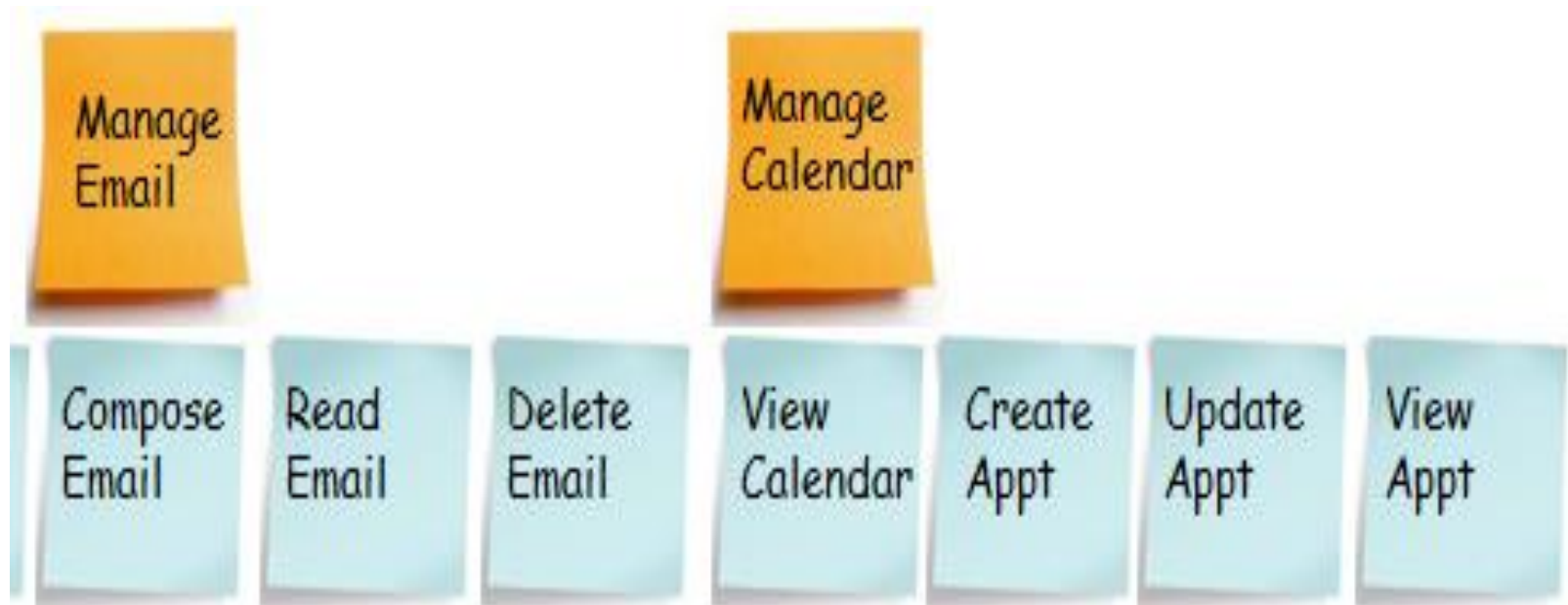
- ✓ Não há regra que determine quanto o projeto deve durar.
- ✓ Para projetos mais longos que quatro meses, uma boa sugestão é usar ciclos trimestrais
 - ✓ Lançar uma versão testada e documentada a cada três meses



Histórias “fechadas”

- ✓ Uma história fechada termina com o alcance de um objetivo de valor. Usuário sente que concluiu algo.
- ✓ Em vez de:
 - ✓ Uma recrutadora pode gerenciar vagas publicadas.
- ✓ Prefira:
 - ✓ Uma recrutadora pode revisar currículos para vagas de um anúncio;
 - ✓ Uma recrutadora pode alterar a data de expiração de uma vaga;
 - ✓ Uma recrutadora pode excluir um currículo não adequada à vaga;
- ✓ Consonância com outras diretrizes sobre tamanho de histórias.

Histórias “fechadas”

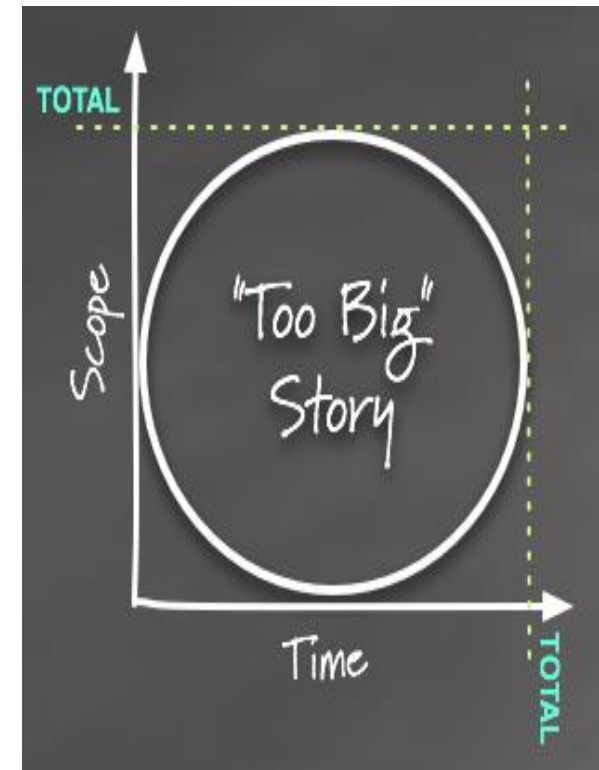


Restrições em cartões

- ✓ Devem ser obedecidas e não implementadas diretamente.
- ✓ Exs:
 - ✓ Sistema deve suportar picos de 50 usuários simultâneos.
 - ✓ O sistema novo deve utilizar o banco de dados existente.
- ✓ Não são alocadas a iterações e não são diretamente estimadas.
- ✓ Podem ser colados na parede.
- ✓ Podem ter testes de aceitação associados.

Níveis de detalhes diferenciados

- ✓ Histórias que serão implementadas nas próximas iterações possuem nível de detalhe maior que histórias que ainda demorarão para ser implementadas.
- ✓ Planejamento em ondas sucessivas.



Por que refinar os requisitos progressivamente?

- ✓ As coisas mudarão
 - ✓ Requisitos com maior probabilidade de mudar devem ser construídos mais à frente.
- ✓ Não há necessidade
 - ✓ Visibilidade é suficiente para os próximos itens apenas (como um farol)
- ✓ O tempo é escasso
 - ✓ Descrever com mais detalhes somente quando for a hora adequada.

Separar solução de histórias


- ✓ Problema comum no universo de requisitos.
- ✓ Evitar principalmente detalhes de interface.
- ✓ Detalhes de interface tendem a aparecer em histórias de modificação e evolução do sistema, pois a interface está pronta.

Algumas coisas não são histórias

- ✓ Formato de histórias de usuário não é apropriado para tudo.
- ✓ Se for necessário documentar requisitos em outro formato, então use formato adequado.
 - ✓ Ex. documentos de interface
- ✓ Documentos de fornecedores externos também podem demandar outros formatos.

Incluir papéis de usuário nas histórias

- ✓ Se papéis de usuário foram identificados, então eles devem aparecer nas histórias de usuário.
- ✓ Diferença é pequena mas o tipo de usuário fica mais presente.
- ✓ Deve-se lembrar do formato sugerido para a escrita de histórias de usuário.

A yellow rectangular box with a thin black border containing the User Story Template text.

User Story Template
As a [user role]
I want to [desired feature]
so that [value/benefit]

Escreva para um usuário

- ✓ Histórias tem leitura mais fácil quando escritas direcionadas a um usuário.
- ✓ Problemas de interpretação ou ambiguidade podem ficar mais evidentes.
 - ✓ Um recrutador pode excluir currículos.
 - ✓ Um recrutador pode excluir seus próprios currículos.

Escreva na voz ativa

- ✓ Histórias de usuário são mais fáceis de ler e entender quando escritas na voz ativa.
- ✓ Em vez de:
 - ✓ Um currículo pode ser publicado por um recrutador.
- ✓ Escreva:
 - ✓ Um recrutador pode publicar um currículo.

Cliente escreve histórias

- ✓ Idealmente cliente escreve histórias.
- ✓ Em alguns projetos, desenvolvedores ajudam, mas responsabilidade final deve ficar com os clientes.
- ✓ Clientes também devem priorizar as histórias.
- ✓ Escrever as histórias aumenta o envolvimento dos clientes.

Não numere os cartões

- ✓ Pode ser uma tentação para manter controle dos cartões ou ter algum mecanismo de rastreabilidade.
- ✓ Acrescenta esforço desnecessário.
- ✓ Normalmente histórias são referenciadas pela funcionalidade em vez de por um número.
- ✓ Se sentir a necessidade, tente acrescentar um título para a história.

Possíveis problemas com histórias

- ✓ Alguns possíveis problemas que podem ocorrer com a elaboração de histórias.
- ✓ Em seguida, serão fornecidas diretrizes para evitar estes problemas com a escrita de histórias de usuário.

INVEST
in good user stories



Histórias pequenas demais

- ✓ Histórias muito pequenas causam problemas de estimativas quando a ordem da implementação afeta consideravelmente as estimativas.
 - ✓ Resultados da pesquisa devem ser salvos em um arquivo XML.
 - ✓ Resultados da pesquisa devem ser salvos em um arquivo HTML.
- ✓ Ao se colocar a história em uma iteração, ela pode ser quebrada, mas antes disso, as histórias deveriam ser combinadas (para efeitos de planejamento).

Histórias dependentes

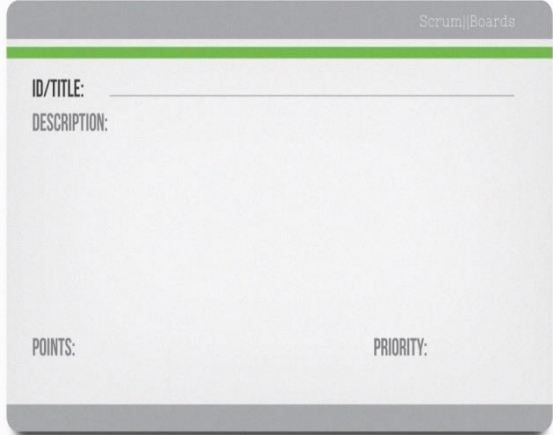
- ✓ Pode haver dificuldade no planejamento das iterações quando a histórias dependentes.
- ✓ A inclusão de uma história em uma iteração acarreta a inclusão de outra relacionada.
- ✓ Pode acontecer quando as histórias são muito pequenas ou foram divididas de maneira inapropriada.
- ✓ Solução pode ser combinar histórias.
- ✓ Lembrar da divisão em camadas (fatias do bolo).

Goldplating (“Dar de bandeja”)

- ✓ Acontece quando são entregues funcionalidades que não foram solicitadas ou além do que é necessário para sua implementação.
- ✓ Difícil impressionar o cliente com a participação constante em projetos ágeis.
- ✓ Pode-se evitar aumentando a visibilidade das tarefas de cada um e reuniões de término de iteração (demonstração do que foi implementado). Pode evitar ocorrências futuras.
- ✓ Uso de uma equipe de Garantia de Qualidade.

Detalhes demais

- ✓ Acontece quando muito tempo é gasto com detalhes antes da implementação da história ou ainda com a escrita da história.
 - ✓ Demonstra algo valor para documentação em detrimento da conversação.
- ✓ Uso de cartões (espaço limitado).



A template for a User Story Card, titled "USER STORY CARD" in large, bold, grey letters at the top. Below the title is a thin green horizontal line. The card has a light grey background with a darker grey border. The text "Scrum|Boards" is visible in the top right corner. The card contains fields for "ID/TITLE:", "DESCRIPTION:", "POINTS:", and "PRIORITY:". The "ID/TITLE:" and "DESCRIPTION:" fields are on the left side, with the "DESCRIPTION:" field being larger. The "POINTS:" and "PRIORITY:" fields are on the right side, with the "PRIORITY:" field being larger.

Incluir detalhes de interface muito cedo

- ✓ Em algum momento, detalhes de interface poderão aparecer nas histórias.
- ✓ Entretanto, este nível de detalhe deve ser deixado para o momento correto no projeto.

Dificuldades de priorização

- ✓ Cliente pode ter muita dificuldade em priorizar histórias.
- ✓ Dificuldade pode ocorrer em função de:
 - ✓ Tamanho. Histórias grandes podem ser mais difíceis de se priorizar.
 - ✓ Por falta de expressão do valor para o negócio.
 - ✓ Ex. O usuário pode visualizar informações de erro em um arquivo de log.
 - ✓ Melhor: Sempre que um erro ocorrer, informações suficientes são fornecidas ao usuário para saber como corrigir o erro.
- ✓ Sempre que possível, usuário deve escrever as histórias.

Responsabilidade do Cliente

- ✓ Cliente não aceita a responsabilidade de escrever e priorizar histórias.
- ✓ Possível solução é reduzir a responsabilidade do cliente para que ele se sinta mais à vontade para tomar decisões.

Iniciar o desenvolvimento sem visão

- ✓ É um erro iniciar o desenvolvimento sem nenhuma visão do produto.
- ✓ Deve haver uma visão disponível para enunciar claramente o cliente selecionado, suas necessidades e atributos críticos
- ✓ Esta visão ajuda a determinar quais recursos deverão ser implementados e garante a criação de um produto útil.

Demorar para lançar o produto

- ✓ Esforços tipo big-bang consomem muito tempo e dinheiro e tem alto risco de fracasso.
- ✓ Comece com produto voltado para um conjunto pequeno de necessidades e forneça a funcionalidade mínima exigida.

Especificação de Requisitos disfarçada

- ✓ Uma especificação disfarçada em Product Backlog é tentadora porque atende nosso desejo de saber todos os requisitos de cara
- ✓ Mas não tem suporte para surgimento de novos requisitos.
- ✓ Não vê requisitos com algo fluido e transitório.
- ✓ Pode ser sinal de relacionamento pouco saudável entre Product Owner e equipe.

Empurrando os requisitos

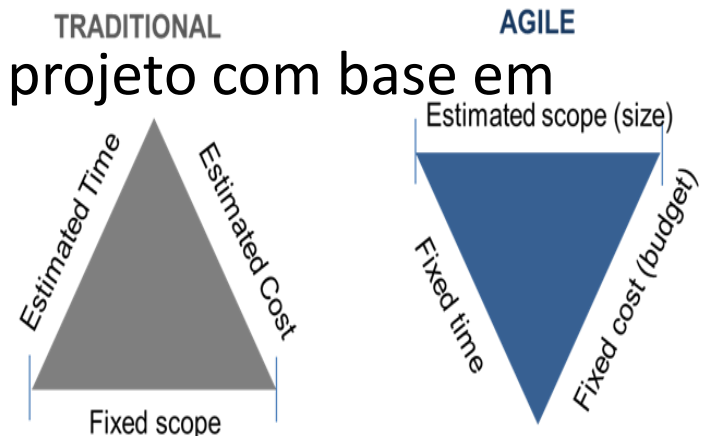
- ✓ Product Owner escrever os itens do backlog sozinho e depois os empurra para a equipe organizar os Sprints.
- ✓ Product Owner deve ser envolver os colegas da equipe.
- ✓ Product Owner deve estar também envolvido nas atividades de planejamento do release.
- ✓ Equipe de negócios e desenvolvedores precisam trabalhar juntos diariamente no decorrer do projeto.

Uso do Sprint Burndown

- ✓ Algumas empresas usam o Sprint Burndown como relatório de projetos nas reuniões de status ou como documento de reporte para gerência superior.
- ✓ Sua finalidade é ajudar a equipe e inspecionar seu progresso e adaptar o trabalho.
- ✓ Não é relatório de status.
- ✓ Usá-lo como relatório o transforma em mecanismo de controle.
- ✓ Stakeholders podem ser convidados para reuniões com a equipe ou outro relatório deve ser usado.

Contratos de Preço Fixo

- ✓ Se for possível, evite projetos com preço e escopo fixos.
- ✓ Se não for possível, divida um contrato de preço fixo em duas partes.
 - ✓ O primeiro cria a visão do produto e implementa-o parcialmente.
 - ✓ O segundo continua a dar vida ao projeto com base em feedback do cliente.



Referências

- Cohn. Mike. Desenvolvimento de software com scrum – Aplicando métodos ágeis com sucesso. Editora Bookman. 2011
- COHN, Mike; User Stories Applied: For Agile Software Development; Addison-Wesley, 2004.
- Pichler, Roman. Gestão de Produtos com Scrum – Implementando métodos ágeis na criação e desenvolvimento de produtos. Editora Campus.2011
- Adkins, Lyssa. Coaching agile teams – A companion for Scrum Masters, Agile Coaches, and Project Managers in transition. Editora Addison Wesley. 2011.