



Engenharia de Requisitos

Tecnologia de Software

Aula 4: História do usuário

Fábio Levy Siqueira

levy.siqueira@usp.br

Representação de requisitos

- Existem diversas formas de representação
 - Lista de requisitos (afirmações)
 - Linguagem natural e linguagem natural estruturada
 - Cenários
 - *Exemplo*: narrativas, casos de uso e histórias
 - Modelos de metas
 - *Exemplo*: i*, Tropos e KAOS
 - Linguagens de especificação formal
 - *Exemplo*: Z, B e Event-B
 - Diagramas em geral
- **Cada uma tem vantagens e desvantagens**

Representação de requisitos

- Requisitos representam fenômenos no ambiente
 - O que os usuários desejam no ambiente com o software
 - Não são detalhes do software



- “Especificação”: fenômenos **compartilhados**
 - A parte do fenômeno que deve ser atendida pelo software
- *Representações descrevem o ambiente*

Representação de requisitos

- Representações não devem tratar de
 - Detalhes do software
 - Detalhes de implementação
 - Detalhes do ambiente que não envolvem o software



- A fronteira não é tão precisa
 - Requisitos dos *stakeholders* / restrições

Representação de requisitos

- Independente do formato, um requisito deve ser (ISO 29148, 2018)
 - Necessário
 - Adequado (nível de abstração adequado)
 - Não ambíguo
 - Completo (descrição suficiente *para o seu uso*)
 - Singular (*depende da representação*)
 - Factível
 - Verificável
 - Correto
 - Em conformidade com o padrão definido

Representação de requisitos

- Características importantes para o modelo como um todo (ISO 29148, 2018)
 - Completo
 - ...não é possível garantir a completude...
 - Mas não deve haver requisitos com "a ser definido"
 - Consistente
 - Factível
 - Compreensível (estar claro o que é esperado pelo sistema)
 - Possível de ser validado

Texto

- Termos a evitar
 - Superlativos (ex.: "melhor" e "pior")
 - Linguagem subjetiva ("amigável" e "rápido")
 - Pronomes vagos (ex.: "isso" e "aquilo")
 - Advérbios e adjetivos ambíguos (ex.: "quase sempre")
 - Termos abertos e não verificáveis
 - Frases comparativas
 - Aberturas (ex.: "se possível" e "conforme apropriado")
 - Referências incompletas (dificulta verificação)
 - Afirmações negativas (confuso)

Exemplo

- Como escrever **requisitos de software** usando linguagem natural?
 - *Exemplo*

A interface deve ser *amigável*.

O usuário deve ser capaz de *gerenciar* os pedidos.

Exemplo

- Um requisito de software precisa ser *verificável e não ambíguo*

A interface deve ser *amigável*.

O usuário deve ser capaz de *gerenciar* os pedidos.

História do usuário

História

- Proposto pela Programação eXtrema (XP)
 - Descreve uma funcionalidade que será de **valor** para o usuário ou comprador do software (COHN, 2004)
 - Perspectiva de **quem quer** a característica
- Premissas
 - Ambiente turbulento / inovação
 - Não é possível definir **todos** os requisitos desde o começo
 - Requisitos e o ambiente mudam
 - Requisitos podem não ser mais necessários
 - Custo de detalhar algo / custo de atualizar
 - Dificuldade de expressar todos os detalhes necessários para os desenvolvedores em um documento

} Desperdício

História

- Conceitos relacionados
 - Épico
 - História grande
 - Ajuda para o planejamento de *release*
 - Histórias que não serão tratadas não precisam ser detalhadas
 - Tarefas
 - Detalhes dos desenvolvedores
 - Como eles irão implementar a história
 - (É opcional)

História

- Conteúdo: 3C
 - Cartão (descrição escrita)
 - “Lembrete de uma conversa”
 - Conversas
 - Entre o desenvolvedor e o *representante do cliente*
 - Imediatamente antes/durante a implementação da história
 - Confirmação
 - Como confirmar que a história está completa
 - Critério de aceitação da história

História não é só o texto (cartão)!

Cartão

- Manifestação da história
 - ...mas é só um lembrete para a conversa...
 - Cartão/post-it

- Naturalmente imprecisas

Como um comprador quero acompanhar o pedido feito para que eu saiba quando os produtos chegarão

- Os clientes / *Product Owners* escrevem as histórias
 - Podem ter ajuda de desenvolvedores
 - Desenvolvedores podem escrever histórias
 - O PO ainda precisará concordar: ele quem irá priorizar!

Cartão

- Existem diferentes formatos
 - Afirmação em linguagem natural

Buscar produtos na loja

- Formato estruturado (COHN, 2004)
 - *Como um* <tipo do usuário>
Eu quero <algum objetivo>
Para que <algum motivo>

Como um comprador

Eu quero buscar produtos disponíveis na loja

Para que eu possa comparar seu preço e compra-los

- No “para que” aparecem **metas** do usuário

Cartão

- Outras possíveis informações (PATTON, 2014)
 - Nome
 - Número
 - Estimativa
 - Dependências
 - Data
- ...naturalmente dependem do formato...
- O formato em si não é tão importante: *ideia*

Cartão

- Detalhes

- O cartão deve ser pequeno
 - É só um lembrete!
- Deve-se evitar detalhes de implementação e interface
 - Não é uma tarefa do desenvolvedor: é algo que o PO consegue dar **valor**
- São continuamente refinadas
 - Histórias grandes são detalhadas
 - (*Grooming* do Scrum)

Conversas

- Permitem que os desenvolvedores entendam os detalhes necessários para implementar

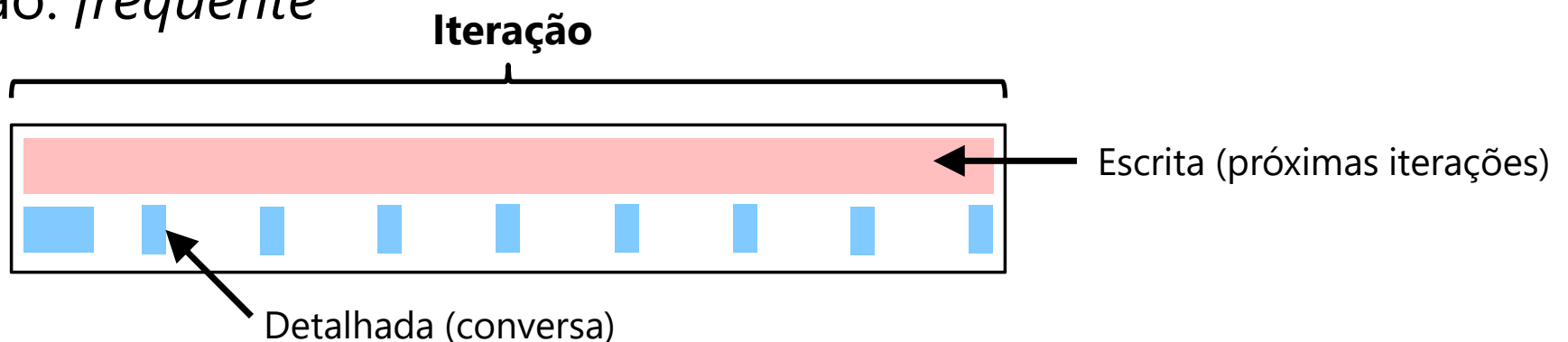
- *Exemplo*

Como um comprador **eu quero** buscar produtos disponíveis na loja **para que** eu possa comparar seu preço e compra-los

- A busca deve consultar só o nome e a descrição?
 - Como é o resultado da busca?
 - O que acontece se nenhum produto for encontrado?
 - O usuário precisa estar logado?
 - ...
- Foco da atividade de requisitos não é a escrita
 - **É a conversa!**

Conversas

- Não é só para detalhar a história
 - Conversa para escrever a história e parte da confirmação (antes da iteração)
 - Conversa para estimar a história
 - Conversa para quebrar histórias em tarefas
 - Conversa para tirar dúvidas sobre a história/tarefa
- Conversas para detalhar a história acontecem durante toda a iteração: *frequente*



Conversas

- Problemas
 - Disponibilidade do representante
 - Conhecimento do representante
 - *Ideal*: usuário ou representante da área de negócio
 - **Gerente do usuário**: não são usuários reais
 - **Vendedores**: foco na venda e não no uso
 - **Especialistas do domínio**: pode não conhecer o dia a dia
 - **Analista de sistema**: precisa conversar com o usuário/cliente (intermediário)
 - Poder de decisão do representante do cliente
 - Representatividade
- É possível ter um time de representantes

Confirmação

- Condições de satisfação da história
 - Capturam como saber se a história foi implementada corretamente
 - **Perspectiva do PO: especificada por ele**
 - Critério de aceitação *básico*
 - Não é a descrição da funcionalidade: é um critério (*algo* a verificar / testar)
 - Não são casos *concretos* de teste
 - Ajudam a refinar a história
 - Anotações de condições relativas a comportamento esperado/casos excepcionais
 - Anotação de detalhes durante a conversa
 - Condições para testar a história
- Originalmente escrita no verso do cartão de história

Confirmação

▪ *Exemplo 1*

Como um comprador **eu quero** pagar a compra com o meu cartão de crédito **para que** eu possa receber os produtos


- Possíveis critério de aceitação
 - Verificar cartões Visa e Mastercard
 - Verificar números de cartão com menos/mais dígitos
 - Verificar números de cartão inexistentes
 - Verificar cartões expirados
 - Verificar valores acima do limite do cliente
 - Verificar valores baixos

Confirmação

▪ Exemplo 2

Como um comprador **eu quero** me cadastrar na loja **para que** eu possa realizar uma compra

● Possíveis critério de aceitação

- Verificar comprador já cadastrado
- Verificar CPF inválido 
- Verificar cliente com produtos já no carrinho de compra
- Verificar cliente sem produtos

Já está entrando em um detalhe: terá CPF. Adequado se for importante para o cliente **ou** se for durante / imediatamente antes da implementação

Confirmação

▪ *Exemplo 2*

Como um comprador **eu quero** me cadastrar na loja **para que** eu possa realizar uma compra

● Critérios de aceitação **ruins**

- Enviar e-mail de confirmação
- Confirmar e-mail
 - São atividades e não condições a serem testadas
 - "Enviar..." pode ser reescrito, mas "Confirmar e-mail" é uma atividade do usuário!
- Cadastrar CPF, e-mail, telefone e endereço
 - Está detalhando as informações necessárias: *especificação*
- Apresentar dados do cliente corretamente ao visualizar perfil
 - Não é uma confirmação desta história – depende de outra
- Cadastrar com CPF 123456789-09, nome "Fábio", ...
 - Muito detalhado: quase um caso de teste

Confirmação

- Quando escrevê-la? (Cohn, 2004)
 1. Em qualquer momento que o desenvolvedor conversa com o PO
 - Durante a *conversa* é de forma mais detalhada
 2. No início da iteração (*Sprint*)
 3. A qualquer momento que se pensa em uma nova condição
 - Durante ou depois do desenvolvimento da história
- Idealmente deveria ser escrita antes de implementar
 - Ajuda a entender a história

Confirmação

- Testes mais detalhados podem ser criados durante a iteração
 - PO com ajuda de um desenvolvedor
 - “Testes de aceitação” → testes funcionais/sistema
- *Acceptance Test-Driven Development* (ATDD)
 - Desenvolvimento dirigido pelos **testes de aceitação**
 - Foco em exemplos de uso do software
 - Testes feitos antes do desenvolvimento
 - Automatizados por ferramentas

Confirmação

- Ferramentas de apoio

- Fitnesse: <http://fitnesse.org/>
- Cucumber: <https://cucumber.io/>



- *Behaviour Driven Development* (BDD)

- Tipo de ATDD: abordagem de ER (SMART, 2015)
- Criado para ajudar a escrever bons testes
 - <https://dannorth.net/introducing-bdd/>
- História em um formato específico
 - *Given-When-Then*

Problemas das histórias do usuário

- Não são uma documentação de requisitos
 - Pode-se perder os detalhes dos requisitos
 - Importância da *confirmação*
 - Pode ser necessária uma documentação adicional

- Necessidade de um PO disponível
 - Tirar dúvidas frequentemente
 - Decisão: o que será implementado
 - Criar, detalhar e priorizar histórias
 - Continuamente
 - Criar as *confirmações*

Problemas das histórias do usuário

- Dificuldade de rastreabilidade
 - Essa história atende a quem?
 - Esse defeito afeta quais histórias?
- Dificuldade de entender relações entre histórias
 - *User Story Mapping* (PATTON, 2014)
- Time pode perder uma visão do todo
- Problemas com times grandes
 - Comunicação é um pré-requisito

Qualidade das histórias e outros detalhes

INVEST

- Características de uma boa história
 - **I**ndependentes
 - **N**egociáveis
 - **V**aliosas para o usuário/cliente
 - **E**stimável
 - **S**mall (pequenas)
 - **T**estável

INVEST

- Independente

- Deve-se **minimizar** as dependências entre as histórias
 - Dificuldade de *estimar e priorizar*
 - *Exemplo*

Como um comprador **eu quero** pagar a compra com o meu cartão *Visa* **para que** eu possa receber os produtos

Como um comprador **eu quero** pagar a compra com o meu cartão *Mastercard* **para que** eu possa receber os produtos

- Uma solução: generalizar para *cartão de crédito*
 - Na confirmação se faria o teste para cada operadora

INVEST

- Negociável
 - História não é um contrato: é um lembrete
 - Não deve ser tão detalhada!
 - Espaço para o PO negociar os detalhes
- Valiosas para o usuário/cliente
 - Importância para que ele priorize
 - **Mesmo histórias técnicas** precisam ter um benefício para usuário/cliente
 - Se a história for muito técnica, ela deveria ser uma tarefa

INVEST

- Estimável
 - Importante para priorizar (**Aula 6**)
 - Problemas
 - História muito grande
 - Quebrar a história em histórias menores
 - Desenvolvedores não tem conhecimento técnico
 - Fazer um estudo (pode-se gerar uma história para isso)
 - Desenvolvedores não tem conhecimento de domínio
 - Conversa com o *Product Owner*

INVEST

- Pequena (*Small*)
 - Tamanho que caiba em uma iteração
 - Alguns dias X maior que a iteração
- História** **Épico**
- Como quebrar uma história?

Como um comprador **eu quero** comprar produtos **para que** eu receba o produto desejado

Como?



Como um comprador **eu quero** buscar produtos **para que** eu possa escolher o produto desejado

Como?



Como um comprador **eu quero** filtrar a busca por preço **para que** eu possa escolher o produto que caiba no meu bolso

INVEST

- Testável
 - Deve ser possível saber se a história foi implementada com sucesso
 - Envolve a confirmação
 - Possível criar confirmações para ela

Requisitos não funcionais

- Como representar requisitos não funcionais
 - Histórias normais
 - Cartões anotados como **restrições**
 - Pode ser difícil organizar
 - *Exemplo*
 - A confirmação do pagamento deve demorar menos do que 5s
 - Defini-los no **critério de pronto** (Scrum)
 - RNF costumam ser ortogonais às funcionalidades
 - *Exemplo*
 - Testar página no Chrome, no Firefox e no Edge

Bugs

- Cohn (2004) sugere criar um cartão para cada Bug
 - Especialmente se o tamanho for similar ao de uma história
 - Se forem simples, podem ser combinados para facilitar a estimativa
 - "Grampear"
- Podem ser usadas diferentes cores de cartões para diferenciar histórias

Nível de detalhe

- Tamanho: duração (RUBIN, 2013)
 - Épico: maior que o tempo de uma *release*
 - Maior que uma iteração
 - História: tamanho em dias
 - Cabe em uma iteração
 - Tarefa: horas
 - Não é uma história: é uma atividade
 - Pode ser feita em um dia
- Histórias são continuamente refinadas
 - Estão em diferentes níveis de detalhe
 - Épicos: histórias para próximas releases
 - Só as histórias importantes são detalhadas

Nível de detalhe

■ Processo

- Não existe um processo único
- Algumas atividades comuns
 - Workshop de histórias: criação das histórias (**Aula 5**)
 - Planejamento da *release*
 - Planejamento da iteração
 - *Grooming* (Scrum)
 - Durante o desenvolvimento
 - Confirmação das histórias da iteração atual / deste *release*
 - (*Grooming*)

} Estimativa, priorização
e detalhamento (**Aula 6**)

Questões

- Papel ou digital?
 - Post-it / cartões X Trello / Jira
 - Papel
 - Fácil de manipular (criar e priorizar)
 - Visível
 - Digital
 - Armazena histórico do projeto
 - Fundamental para times distribuídos / pandemia

- Jogar fora ou armazenar ao terminar?
 - Prazer de amassar / queimar cartões
 - Pode ser útil para o histórico
 - Mas não é suficientemente detalhado...

Conclusão

- Histórias não são só o texto
 - Confirmação e, principalmente, a **conversa**
- Histórias evoluem com o tempo
 - Nível de abstração (épico)
 - Confirmação
 - Em conversas pode-se identificar novas histórias
- Evita o desperdício de detalhar algo que não será *necessariamente* implementado

Referências

- COHN, M. **User Stories Applied**. Addison-Wesley, 2004.
- ISO. **Systems and software engineering - Life cycle processes – Requirements engineering**. ISO/IEC/IEEE 29148, 2018.
- PATTON, J. **User Story Mapping**. O'Reilly, 2014.
- RUBIN, K. **Essential Scrum: A Practical Guide to the Most Popular Agile Process**. Addison Wesley, 2013.
- SMART, J. F. **BDD in Action**. Manning, 2015.