Engenharia de Requisitos

Tecnologia de Software

Aula 6: Priorização e estimativa

Fábio Levy Siqueira
levy.siqueira@usp.br

Priorização

- É natural que nem todos os requisitos sejam implementados
  - Limitações no projeto
  - ...talvez nem precisem: 2/3 das *features* são raramente ou nunca usadas...

(Standish Group, 2009 apud Wiegers e Beaty, 2013)

- Alguns requisitos precisam ser implementados antes de outros
  - Entregar um valor para o negócio mais rápido

- Necessário **priorizar**
  - Definir a importância do requisito

Priorização

- Prioridade pode ser usada para definir a ordem de
  - **Implementação**
  - Detalhamento
  - Validação
  - Documentação

- Diferentes itens podem ser priorizados
  - Metas, requisitos, cenários, itens de *backlog*

  É importante priorizar itens no mesmo nível de abstração

- A prioridade pode mudar durante o projeto

Critério

- Qual o critério para priorizar?

Critério

- Questões a considerar independente do critério (Davis, 2005 apud Wiegers; Beaty, 2013)
  - Necessidade dos clientes
  - A importância relativa dos requisitos para o cliente
  - O *timing* para entregar a *feature*
  - A dependência entre os requisitos
    - ...e a importância para a arquitetura
  - Necessidade de implementação de requisitos em conjunto
  - Custo de satisfazer cada requisito

Técnicas

- Algumas técnicas (Pohl, 2011 e Wiegers; Beaty, 2013)
  - *Ad hoc*
  - "Dentro ou fora" (*in or out*)
  - Três níveis
  - Classificação de Kano
  - MoSCoW
  - $100

  - (Existem algumas outras)

Técnicas

- *Ad hoc*
  - Priorizado por um ou um grupo de *stakeholders* baseado em um ou mais critérios
  - Pode ser organizado em grupos
    - Requisitos com prioridade similar ficam no mesmo grupo
  - Em geral é usada no Scrum
    - Valor para o cliente e "custo"

- "Dentro ou fora"
  - Decisão binária: dentro ou fora desta *release*/do escopo
  - Decisão de um grupo de stakeholders

Técnicas

- Três níveis
  - Alta, média e baixa
  - *Problema*: todos os requisitos com prioridade alta
  - Uma outra forma: importância e urgência
    - Importância: clientes precisam dessa funcionalidade
    - Urgência: clientes precisam disso para o próximo *release*

	Importante	**Não importante**
Urgente	Alta prioridade	Não fazer
Não tão urgente	Prioridade média	Baixa prioridade

Se não é importante, ou dê baixa prioridade ou despreze-o

(Wiegers; Beaty, 2013)

Técnicas

- Classificação de Kano
  - "Que gera insatisfação" (*dissatisfier*)
    - Propriedades **básicas** de satisfação
    - O sistema precisa ter para ser usado
  - "Que satisfaz" *(satisfier)*
    - Propriedades **esperadas** de satisfação
    - Cliente demanda conscientemente o requisito
  - Que traz contentamento (*delighter*)
    - Propriedades **inesperadas** de satisfação
    - Cliente não está ciente do requisito / não espera no sistema
      - Surpreende positivamente
  - (É proposto um método sistemático para classificar)

Técnicas

- MoSCoW
  - **Must**
    - A solução tem que ter (obrigatório para o sucesso)
  - **Should**
    - É importante para a solução, mas não é obrigatório para o sucesso
  - **Could**
    - Desejado, mas pode ser adiada ou eliminada
  - **Won’t**
    - Não precisa ser implementada nesta versão

Técnicas

- $100
  - Membros do projeto tem $100 imaginários para gastar com os requisitos
    - Alocam a quantia desejada em cada requisito
  - Prioriza-se pelo dinheiro alocado
  - *Problemas*
    - As últimas pessoas que alocam podem "jogar" com as prioridades
    - Alguém alocar $100 em um único requisito
    - Não considera o esforço necessário

Estimativa

Estimativa

- Dependendo da técnica de priorização pode ser importante **estimar** o requisito
  - Tamanho, complexidade e duração
  - Informação para se considerar ao priorizar
  - Fundamental para o planejamento: duração do projeto

- Atividade de gerência de projetos ligada a ER
  - Planejamento

- Fundamental para o Scrum
  - Estimar itens de backlog / tarefas

Estimativa no Scrum

- Estimativa deve ser feita pelo time
  - Por quem irá implementar
  - (Não pelo PO, Scrum Master ou Gerente)

- Importância da discussão: história do usuário
  - Forma de entender melhor os requisitos

- Estimativas não são um compromisso
  - Devem ser uma medida realista
    - Usada só para o planejamento
  - Não devem ser inflacionadas por questões externas

Estimativa no Scrum

- O que estimar?
  - História
    - Backlog do produto
      - Épicos
    - Backlog da sprint
  - Tarefas
    - Sprint
- Qual métrica usar?
  - Ponto / Hora / dia / semana de trabalho ideal → *Tarefa / História*

- **O que vocês usam?**

Unidade de medição

- Hora / dia / semana de trabalho **ideal**
  - Estimativa de esforço: não é o tempo real
    - Reuniões, outras atividades e interrupções em geral
  - Vantagens
    - Valor concreto
    - Facilita a planejamento
  - Desvantagens
    - Não é real: problema de interpretação
      - Atividade de 1h pode demorar 3h

Unidade de medição

- Ponto de história
  - Considera complexidade, tamanho e outros fatores
  - Vantagens
    - Valor relativo: outras histórias
    - Acurácia X Precisão
    - Fácil de estimar
  - Desvantagens
    - Naturalmente imprecisa
    - Pode não ter significado
    - Depende do time
      - Não é possível comparar pontos de história

Unidade de medição

- O que significa 1 ponto?
  - Algumas alternativas
    - 1 ponto = 1 dia de trabalho ideal (Cohn, 2004)
    - História simples (referência)
    - Menor história do projeto

- Escala
  - Tipicamente a sequência de Fibonacci modificada
    - 1, 2, 3, 5, 8, 13, 20, 40 e 100
    - Foco na acurácia: imprecisão natural
  - Para histórias no backlog do produto
    - “Tamanho de camisas”: P, M e G

Unidade de medição

- Detalhes
  - A soma das histórias quebrada a partir de um épico não precisa ser o valor do épico
  - É comum se triangular o valor das estimativas
    - Maior que a história X e menor que a história Y

- Em geral a estimativa é feita através do *Planning Poker*

Planning Poker

- Variação da técnica *Wideband Delphi*
- Características
  - Baseada em consenso
    - Discussão
  - Usa opinião de especialistas
    - Time de desenvolvimento
  - Tamanho relativo

- Cartões
  - Sequência de Fibonacci modificada
  - Dúvida/Infinito
  - Café

Planning Poker

- Funcionamento
  1. *Product Owner* apresenta um item
  2. Time de desenvolvimento tira dúvidas sobre o item
  3. Cada estimador escolhe uma carta
  4. Todos mostram as cartas ao mesmo tempo
  5. Se houver consenso, o valor é a estimativa
  6. Se não houver consenso
     - Os membros do time discutem a estimativa
       - *Em geral*: quem fez a menor e a maior estimativa justificam o motivo
     - Volta-se ao passo 3

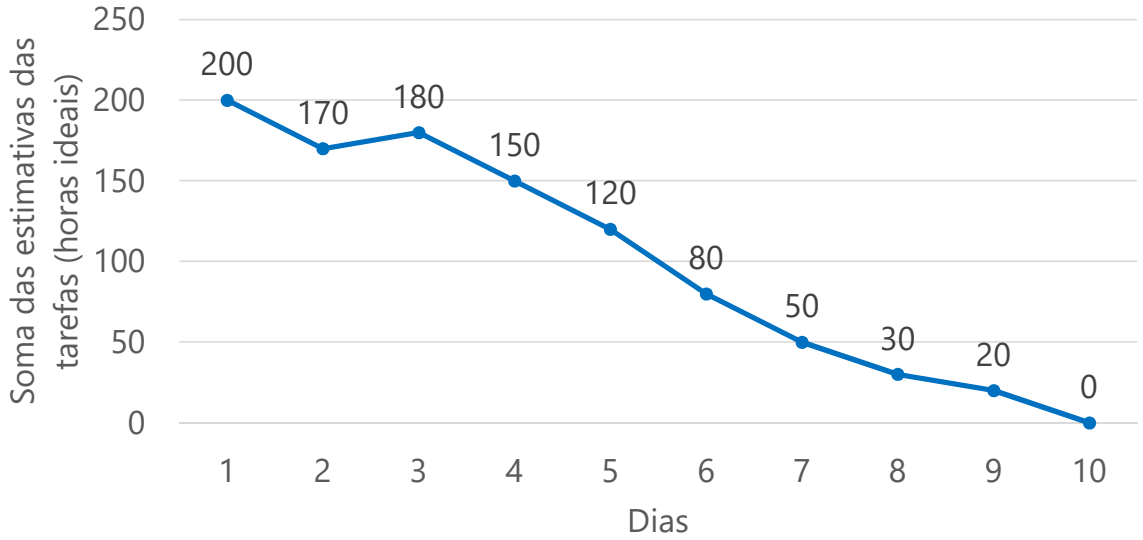
Velocidade

- Quantidade de trabalho **feito** em uma Sprint
  - Soma das estimativas das histórias
  - *Exemplo*
    - 40 pontos de história
    - 40 horas ideais

- Mede a saída do time
  - Apenas histórias **prontas**
    - Definição de pronto (DoD)

Velocidade

- Uso
  - Planejamento
  - Métrica de processo: melhoria
  - Acompanhamento: *burndown* chart



Velocidade e Planejamento

- Planejamento da *release*
  - Conjunto de Sprints
    - Trata de um tema
  - Previsão do número de Sprints para fazer a *release*
    - Pode ser usada como faixa
      - Entre 35 e 40 pontos por Sprint: entre 2 e 3 sprints

- Planejamento da Sprint
  - Quantas histórias podem ser implementadas
    - Capacidade em horas / pontos de história
  - Pode ser um planejamento em uma ou duas partes

Velocidade e Planejamento

- Planejamento em duas partes
  1. PO e time acordam as histórias até a capacidade
  2. Time quebra histórias em tarefas e estima as tarefas
  3. Compara-se a estimativa de tarefas à capacidade
     - Revê histórias: acima ou abaixo da estimativa
     - Troca histórias

- Planejamento em uma parte
  1. Enquanto houver capacidade, escolhe-se um item e ganha-se confiança sobre a estimativa
     - Quebra-se em tarefas / tira-se dúvidas
  2. Termina-se quando não há mais capacidade

Velocidade e Planejamento

- Baseada em dados históricos
  - Fácil quando já se tem várias Sprints
  - Como definir a capacidade para a 1ª Sprint?
    - Quantidade de histórias que o time considera adequado
      - Pode-se balancear com dados de outros times

- Cuidados
  - "Inflação de pontos"
    - Pontos de história não significam nada
  - Velocidade tende a ser estável
    - Mudanças no processo podem afetar
    - Velocidade inicial é mais lenta (conhecimento técnico/do projeto)

Estimativa e priorização

- Quando estimar e priorizar?
  - *Refinamento*: criar, refinar, estimar e priorizar itens
    - Tipicamente histórias
  - Atividade colaborativa do *Product Owner*, *stakeholders*, Scrum Master e desenvolvedores

- Pode acontecer em vários momentos
  - Planejamento da release
  - Conversas do PO com stakeholders durante o projeto
    - (Criar / priorizar itens)
  - Workshop semanal do PO com um ou mais devs
  - Diariamente

Estimativa e priorização

- Definição de "preparado" (*ready*)
  - Similar à definição de pronto (DoD x DoR)
  - Checklist para dizer se um item está preparado para ser usado no planejamento da Sprint

  - *Exemplo* (Rubin, 2013)
    - Detalhes são entendidos pelo time
    - Dependências foram identificadas e não existem dependências externas que afete o item
    - Estimado para caber em uma Sprint
    - Critérios de aceitação definidos e testáveis

Conclusão

- Qual técnica de priorização usar?
  - Depende do processo e do projeto
    - Projetos pequenos podem usar técnicas informais
    - Projetos grandes podem ter que usar técnicas mais formais

- Em métodos ágeis
  - Prioridade é definida pelo PO
    - Baseada nas estimativas do time de desenvolvimento
    - Time pode indicar dependências / impactos na ordem / importância arquitetural

Aula

Projeto integrado

- Priorização
  - Ignorem dependências entre *features*
    - Cadastramento pode ser manual (interface simples / direto no BD)
    - Pode ser usado um valor padrão necessário

- Corrijam histórias e priorizem as 5 histórias definidas na aula passada
  - Estimem essas histórias usando *planning poker*
    - https://www.planitpoker.com/
  - Quebrem a história se necessário

Projeto integrado

- Definição de preparado
  - Texto no formato de história
  - Estimado para caber em uma Sprint
  - Confirmação definida e testável (abstratos – sem valores)

- Para a entrega
  - Definir histórias (cartão e critério de aceite) para as *features*
    - Grupos de 3 pessoas 15 histórias
    - Grupos de 4 pessoas 20 histórias
    - Grupos de 5 pessoas 25 histórias

    } Não deixe *features* sem história
  - Defina épicos (sem critério de aceite) para as demais *features*

Referências

- COHN, M. **User Stories Applied**. Addison-Wesley, 2004.
- POHL, K. **Requirements Engineering: Fundamentals, Principles, and Techniques**. Springer, 2010.
- RUBIN, K. **Essential Scrum: A Practical Guide to the Most Popular Agile Process**. Addison Wesley, 2013.
- WIEGERS, K.; BEATTY, J. **Software Requirements**. 3ª edição. Microsoft Press, 2013.