O Processo de Desenvolvimento de Software

Objetivos

- Contextualizar Análise e Projeto de software dentro de uma metodologia de desenvolvimento (um processo de desenvolvimento de software)

Um processo de desenvolvimento de software

- Uma linguagem de modelagem não é suficiente
- Precisamos também de um processo de desenvolvimento
  - Linguagem de modelagem + processo de desenvolvimento = método (ou metodologia) de desenvolvimento
- O que é um processo de desenvolvimento?
  - Define quem faz o que, quando e como, para atingir um certo alvo
- Veremos os detalhes de processos concretos em outras disciplinas
  - Aqui, só uma introdução
- As grandes fases de qualquer processo de desenvolvimento
  - Planejamento e elaboração
    - Planejamento, definição de requisitos, construção de protótipos (opcional)
  - Construção do sistema (inclui codificação e testes)
  - Implantação (colocar em produção, treinar usuários, ...)

A Fase de Planejamento e Elaboração

1. Criar relatório inicial de investigação (para construir o

business case)

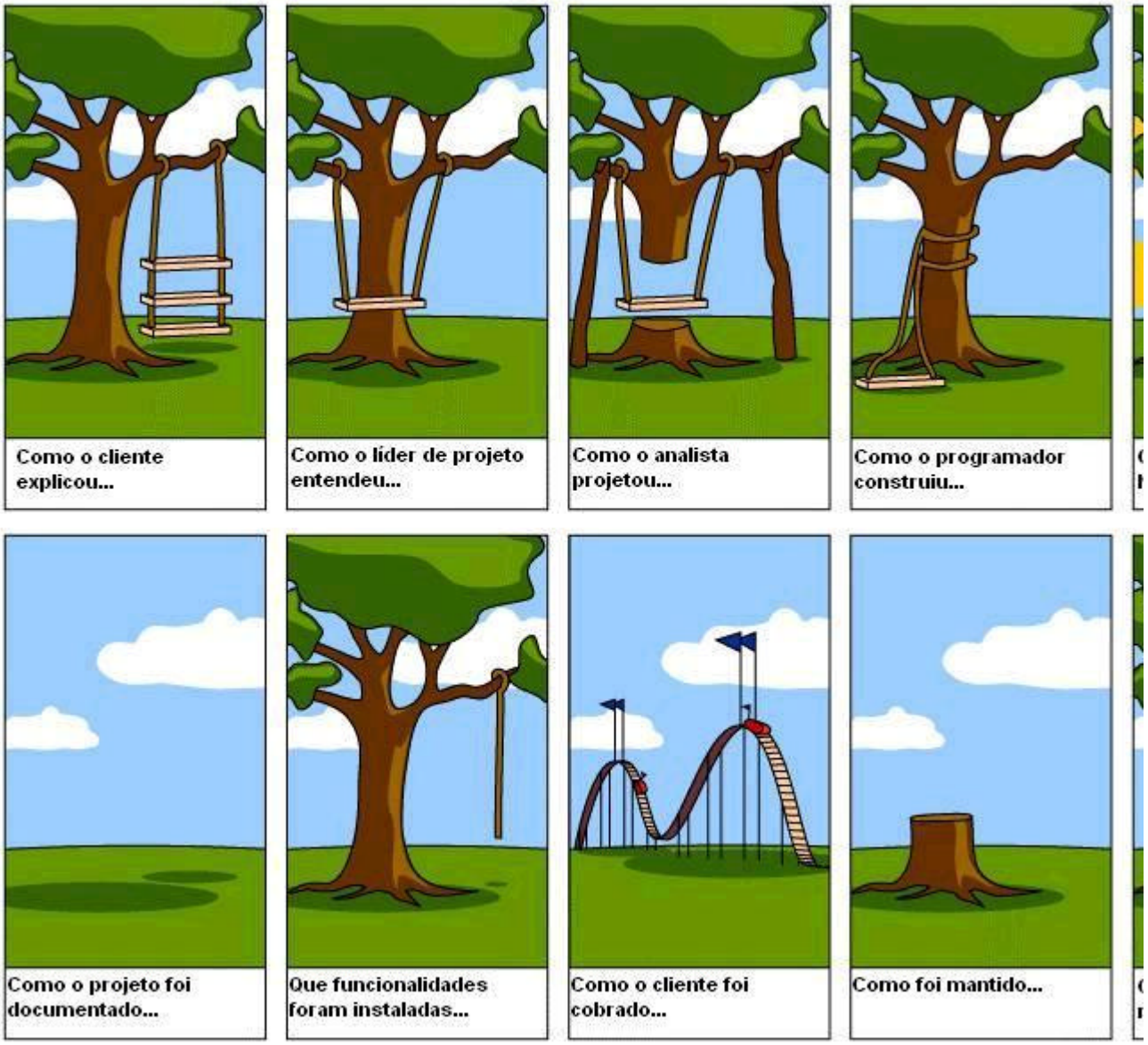
2. Levantar requisitos funcionais e não funcionais
3. Construir glossário (ao longo da fase)
4. Definir modelo conceitual inicial (análise inicial)
5. Projetar arquitetura
6. Priorizar a funcionalidade e distribuí-la entre as iterações

Detalhes sobre o levantamento de requisitos

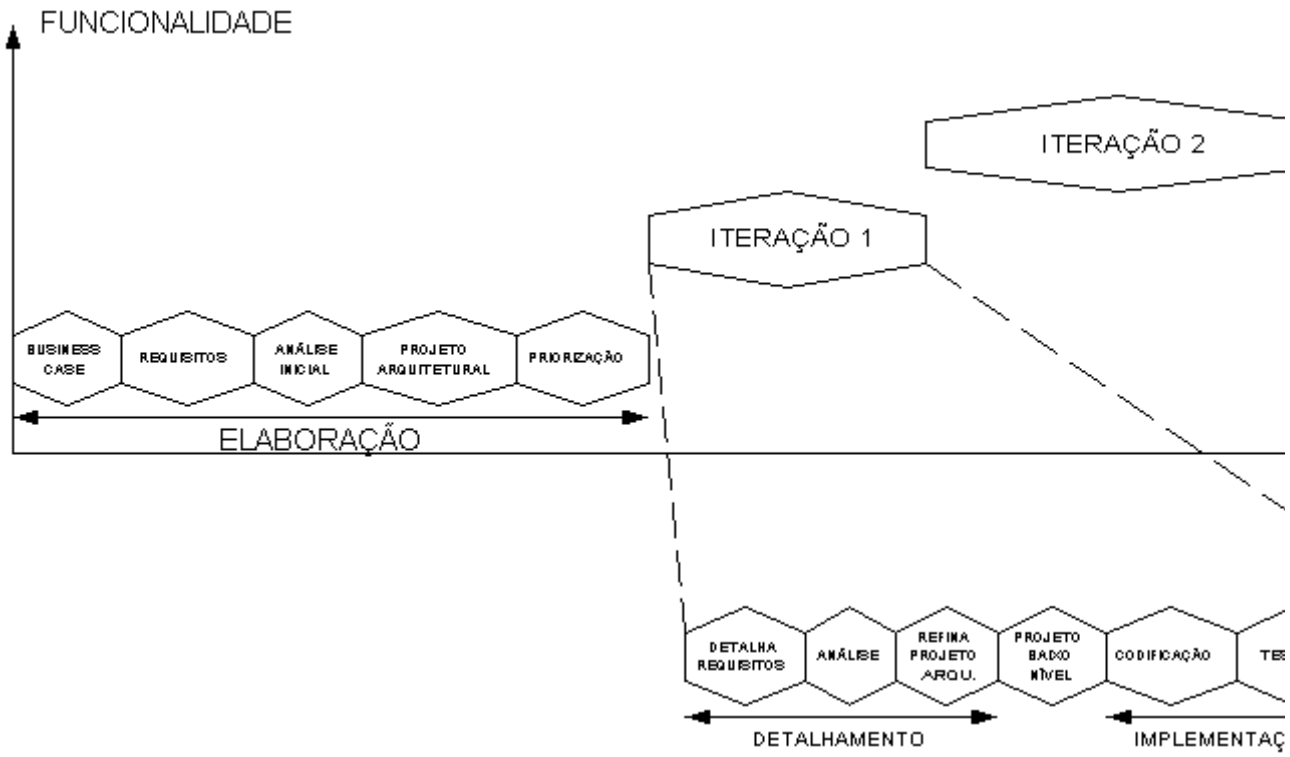
- Requisitos são "cortes" no espaço de solução
- Entendimento do que o usuário quer
- O resultado é uma promessa para o cliente
- Não só requisitos funcionais, mas também:
  - Facilidade de uso necessária
  - Quem utilizará o produto
  - Hardware e software alvo para o produto
  - Qualidade/robustez
  - Desempenho
  - Segurança
  - Compatibilidade com outros produtos/versões e necessidades de migração
  - Necessidades de internacionalização do produto
  - Suporte
  - Preço da solução
  - Documentação necessária
  - Uso de padrões
  - Aspectos legais
  - Integração com outros produtos
  - Packaging
  - etc.
- Não se fala "como" as coisas serão feitas
- "Use cases" descrevem cenários de funcionalidade desejada
  - Também chamados de "User Stories", pois é o usuário que decide o que deve ser feito

Detalhes sobre a fase de Construção

- Hoje, é considerado errado ter um processo que gere um "big bang!"
  - Não se deve ter o software inteiro funcionando por inteiro no primeiro release
  - O risco é grande demais!
- Um processo de desenvolvimento deve ser:
  - Iterativo (ter várias iterações no tempo)
  - Incremental (gerar novas versões incrementadas a cada release)
  - Uma iteração dura entre 2 semanas e 2 meses
- Motivos:
  - Sempre tem algo para entregar para o cliente apressado (a última iteração)
  - Os requisitos mudam com tempo e um processo iterativo mantém freqüentes contatos com o cliente o que ajuda a manter os requisitos sincronizados
  - Altamente motivador para a equipe de desenvolvimento (e o cliente) ver o software funcionando cedo
  - Para evitar isso:



- O que é feito a cada iteração?
  - Análise (refinamento de requisitos, refinamento do modelo conceitual)
  - Projeto (refinamento do projeto arquitetural, projeto de baixo nível)
  - Implementação (codificação e testes)
  - Transição para produto (documentação, instalação, ...)



Detalhes sobre a análise

- A análise gera um modelo para entender o domínio do problema
- Análise também trata em alto nível de como uma solução possível pode ser montada para atender aos requisitos
  - Acaba gerando uma especificação, mas sempre do ponto de vista do usuário e tratando apenas do domínio do problema
- Não trata de detalhes de implementação
- Objetos tratados são sempre do domínio do problema (business objects)
- Muitos diagramas UML podem ser usados
  - O modelo é para o cliente e não para o programador
- Atividades típicas durante a análise

1. Refinar use cases
2. Refinar modelo conceitual
3. Refinar glossário

4. Definir diagramas de seqüência (opcional)
5. Definir contratos de operação (opcional)
6. Definir diagramas de estado (opcional)

Detalhes sobre o projeto (design)

- O projeto é uma extensão do modelo de análise visando sua implementação num computador
- Novos objetos aparecem, mas não são do domínio do problema
- O resultado é para o programador ver, não o cliente
- Objetos da análise são (geralmente) mantidos e são embutidos numa infra-estrutura técnica
  - As classes técnicas ajudam os business objects a:
    - Serem persistentes
    - Se comunicarem
    - Se apresentarem na interface do usuário
    - Terem desempenho aceitável (usando caches ou threads, por exemplo)
- As atividades de projeto incluem:
  - Fase de refinamento da arquitetura (high-level design)
    - Definição de pacotes (módulos), interfaces entre pacotes
    - Decisão sobre uso/criação de bibliotecas e/ou componentes
    - Falaremos disso em detalhes adiante
  - Fase de projeto detalhado (low-level design)
    - Atribuição de responsabilidades entre os objetos
    - Construção de diagramas de classes
      - Pode incluir documentação javadoc (ideal)
    - Construção de diagramas de interação (opcional)
    - Levantamento de necessidades de concorrência
    - Considerações de tratamento de falhas
    - Detalhamento do formato de saída (interface com usuário, relatórios, transações enviadas para outros sistemas, ...)

- Definição do esquema do BD
- Mapeamento de objetos para tabelas se o BD for relacional

Detalhes sobre a implementação

- Escrita do código
- Relativamente simples se o projeto tiver sido bem feito
- Programadores devem normalmente seguir regras de codificação da empresa
- Atividades incluem code reviews
- Não se deve chegar a esta fase cedo demais!
  - Mais cedo você agarra o teclado, mais vai demorar a terminar!
- Poucos novos diagramas nesta fase

Detalhes sobre os testes

- Inclui várias fases de testes
- Testes feitos pelo próprio programador durante a programação
  - Unit test: teste de classes individuais (ou de grupos de classes relacionadas)
  - Functional test: teste de funções inteiras (item de menu, p. ex.)
  - Component test: teste de componentes inteiros (exe, dll, ...) sem (ou com pouco) scaffolding
- Testes feitos por equipes independentes de teste
  - System test: testa a integração entre todos os componentes do produto
  - Alpha test: teste de produto inteiro dentro de casa
  - Beta test: teste de produto inteiro fora de casa
- Testes devem ser automatizados

intro programa