Princípios de Análise e Projeto de Sistemas com UML

3ª edição (2015)

Eduardo Bezerra

Editora Campus/Elsevier

Capítulo 2

Processo de Desenvolvimento de Software

"Quanto mais livros você leu (ou escreveu), mais as aulas você assistiu (ou lecionou), mais linguagens de programação você aprendeu (ou projetou), mais software OO você examinou (ou produziu), mais documentos de requisitos você tentou decifrar (ou tornou decifrável), mais padrões de projeto você aprendeu (ou catalogou), mais reuniões você assistiu (ou conduziu), mais colegas de trabalho talentosos você teve (ou contratou), mais projetos você ajudou (ou gerenciou), tanto mais você estará equipado para lidar com um novo desenvolvimento." - Bertrand Meyer

“Software is hard…”

- Porcentagem de projetos que terminam dentro do prazo estimado: 10%
- Porcentagem de projetos que são descontinuados antes de chegarem ao fim: 25%
- Porcentagem de projetos acima do custo esperado: 60%
- Atraso médio nos projetos: um ano.

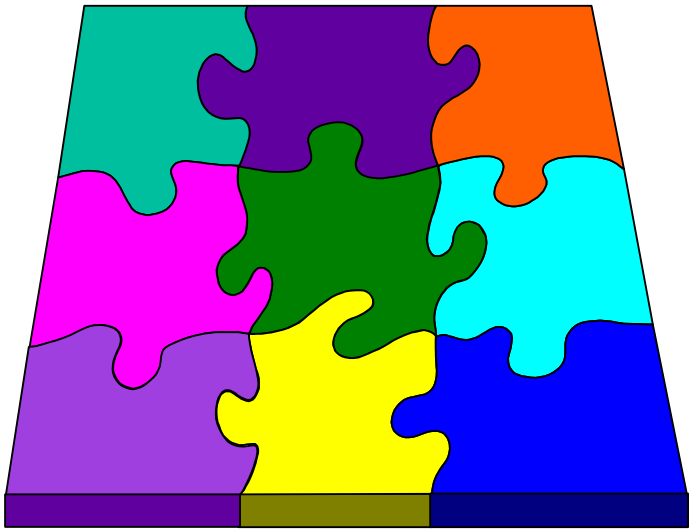
Fonte: Chaos Report (1994)

Processo de desenvolvimento

- Tentativas de lidar com a complexidade e de minimizar os problemas envolvidos no desenvolvimento de software envolvem a definição de ***processos de desenvolvimento de software***.
- Um processo de desenvolvimento de software (PDS) compreende todas as atividades necessárias para definir, desenvolver, testar e manter um produto de software.

Processo de desenvolvimento

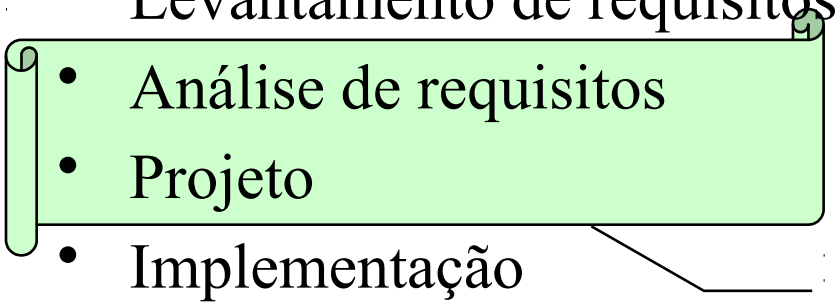
- Exemplos de processos de desenvolvimento existentes:
  - ICONIX
  - RUP
  - EUP
  - XP
  - OPEN
- Alguns objetivos de um processo de desenvolvimento são:
  - Definir *quais* as atividades a serem executadas ao longo do projeto;
  - Definir *quando*, *como* e por *quem* tais atividades serão executadas;
  - Prover pontos de controle para verificar o andamento do desenvolvimento;
  - Padronizar a forma de desenvolver software em uma organização.



2.1 Atividades típicas de um PDS
2.2 O componente humano em um PDS

Atividades típicas de um PDS

- Levantamento de requisitos
- Análise de requisitos
- Projeto
- Implementação
- Testes
- Implantação



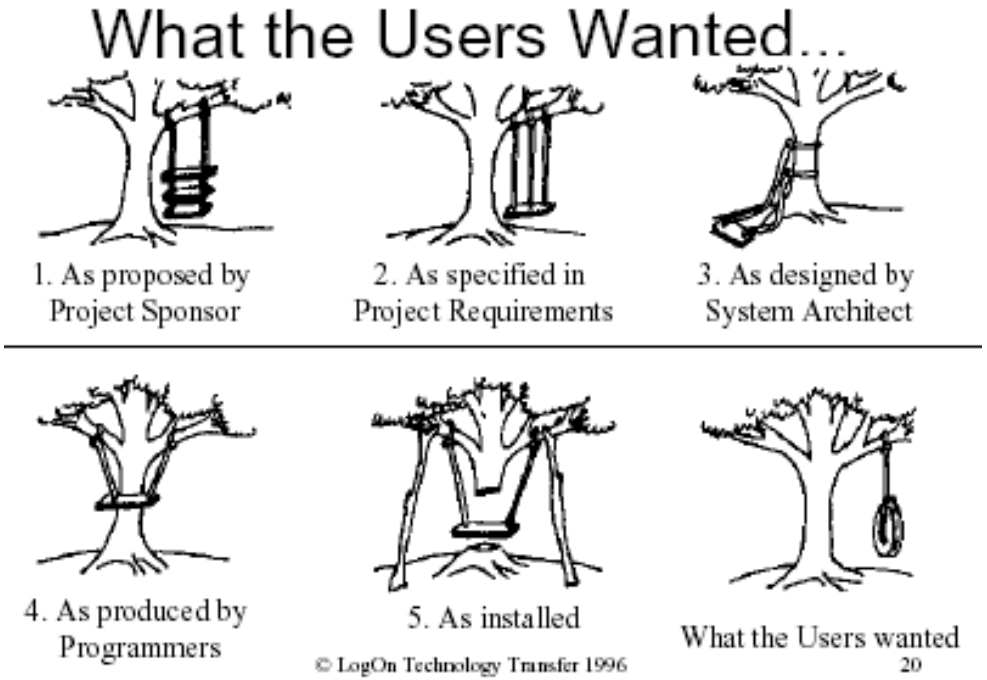
Foco do livro

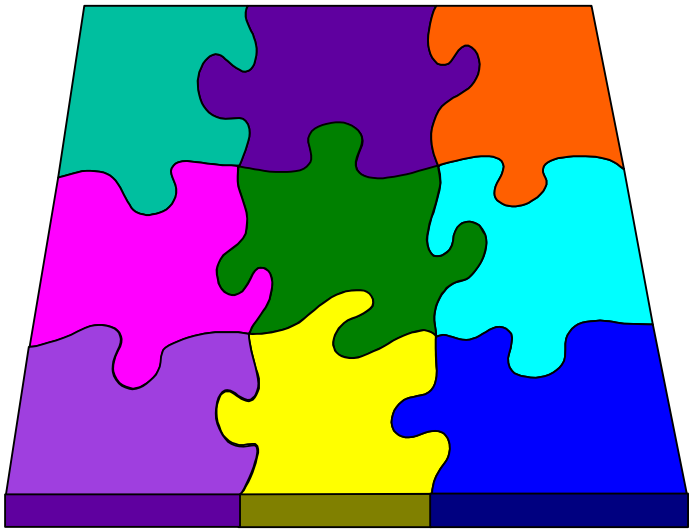
Participantes do processo

- Gerentes de projeto
- Analistas
- Projetistas
- Arquitetos de software
- Programadores
- Clientes
- Avaliadores de qualidade

Participação do usuário

- A participação do usuário durante o desenvolvimento de um sistema extremamente é importante.





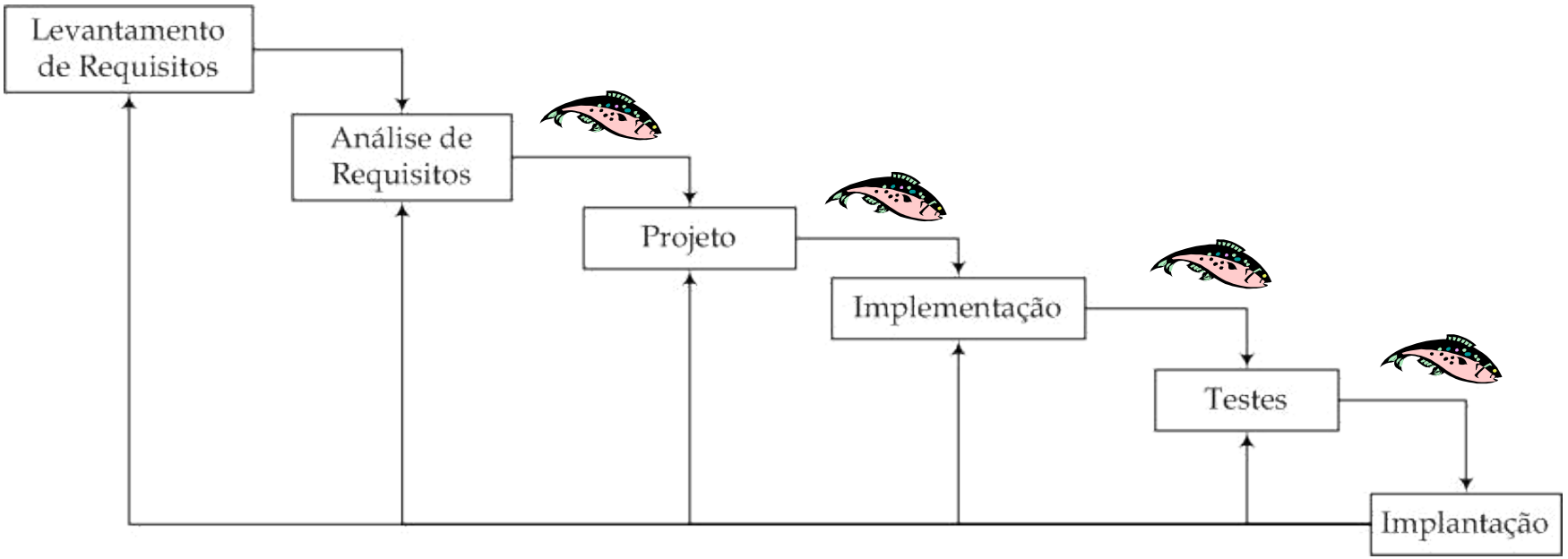
2.3 Modelos de ciclo de vida

Modelo de ciclo de vida

- Um ciclo de vida corresponde a um encadeamento específico das fases para construção de um sistema.
- Dois modelos de ciclo de vida:
  - *modelo em cascata*
  - *modelo iterativo e incremental*.

Modelo em cascata

- Esse modelo apresenta uma tendência para a progressão seqüencial entre uma fase e a seguinte.



Modelo em cascata

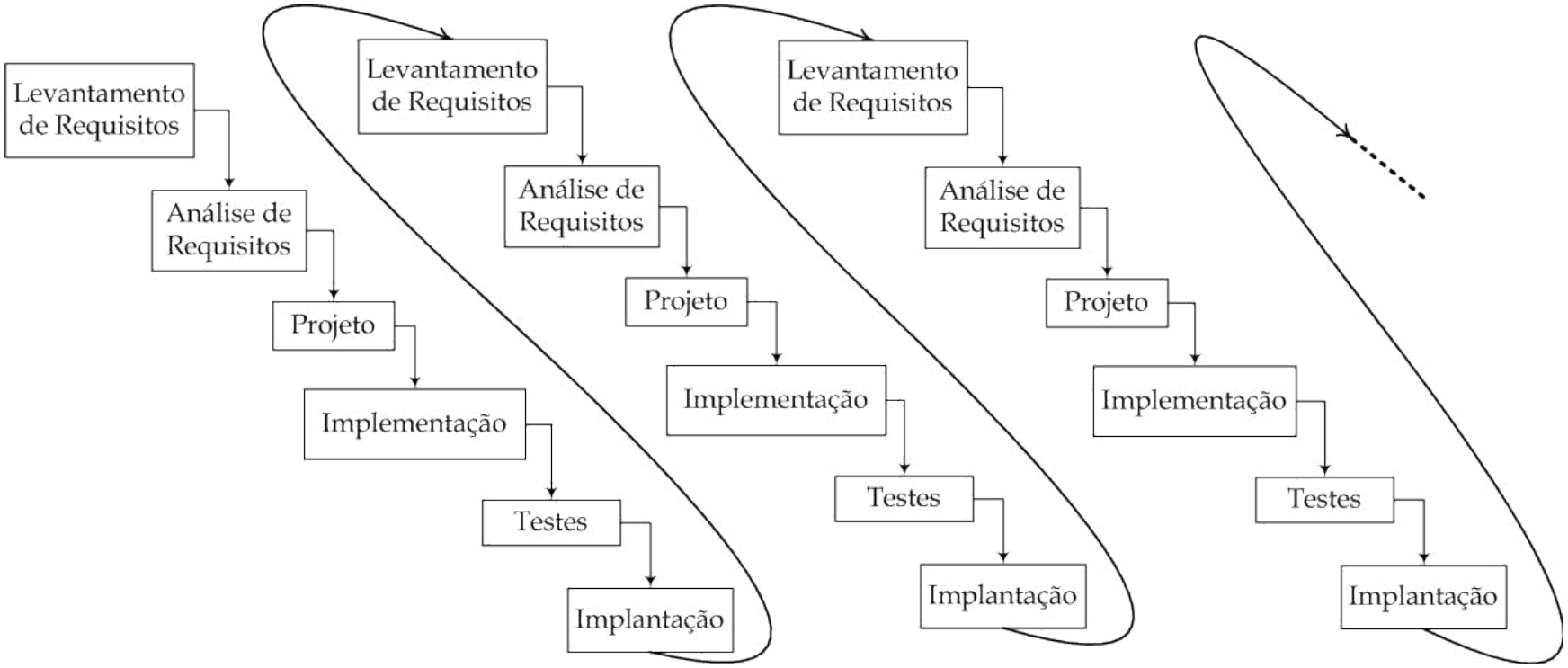
- Projetos reais raramente seguem um fluxo seqüencial.
- Assume que é possível declarar detalhadamente todos os requisitos antes do início das demais fases do desenvolvimento.
  - propagação de erros pelas as fases do processo.
- Uma versão de produção do sistema não estará pronta até que o ciclo do projeto de desenvolvimento chegue ao final.

Modelo de iterativo e incremental

- Divide o desenvolvimento de um produto de software em ***ciclos***.
- Em cada ciclo de desenvolvimento, podem ser identificadas as fases de análise, projeto, implementação e testes.
- Cada ciclo considera um subconjunto de requisitos.
- Esta característica contrasta com a abordagem clássica, na qual as fases são realizadas uma única vez.

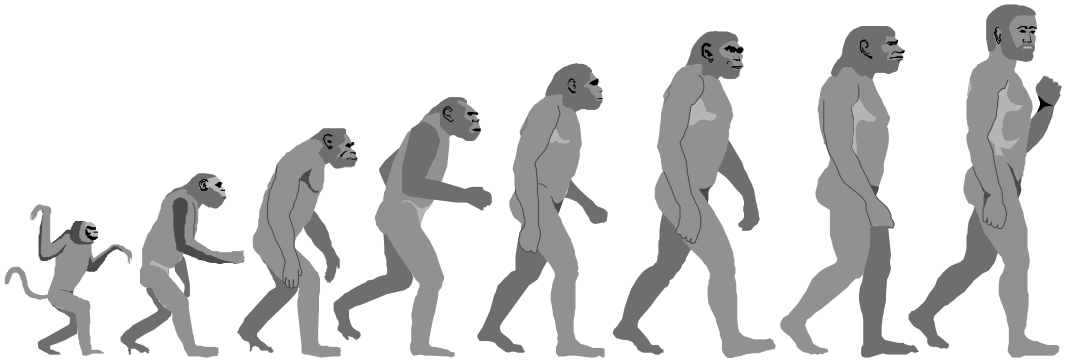
Modelo iterativo e incremental

- Desenvolvimento em “mini-cascatas”.



Modelo iterativo e incremental

- ***Iterativo**:* o sistema de software é desenvolvido em vários passos similares.
- ***Incremental**:* Em cada passo, o sistema é estendido com mais funcionalidades.

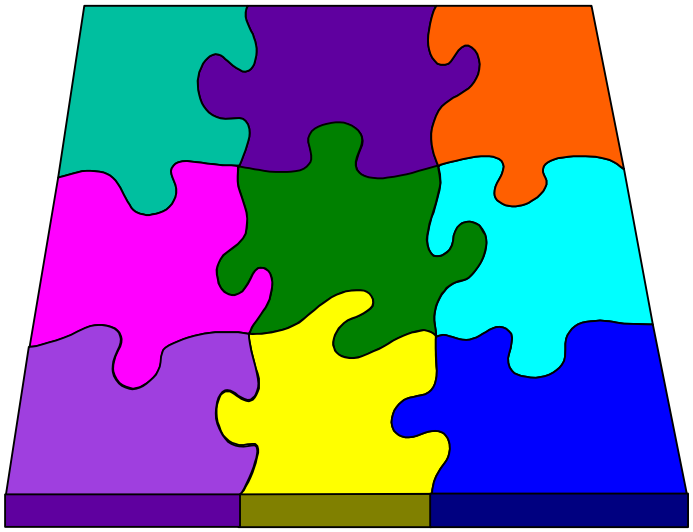


Modelo iterativo e incremental – vantagens e desvantagens

- 👍 Incentiva a participação do usuário.
- 👍 R*iscos* do desenvolvimento podem ser mais bem gerenciados.
  - Um ***risco de projeto*** é a possibilidade de ocorrência de algum evento que cause prejuízo ao processo de desenvolvimento, juntamente com as conseqüências desse prejuízo.
  - Influências: custos do projeto,cronograma, qualidade do produto, satisfação do cliente, etc.
- ☹ Mais difícil de gerenciar

Ataque os riscos

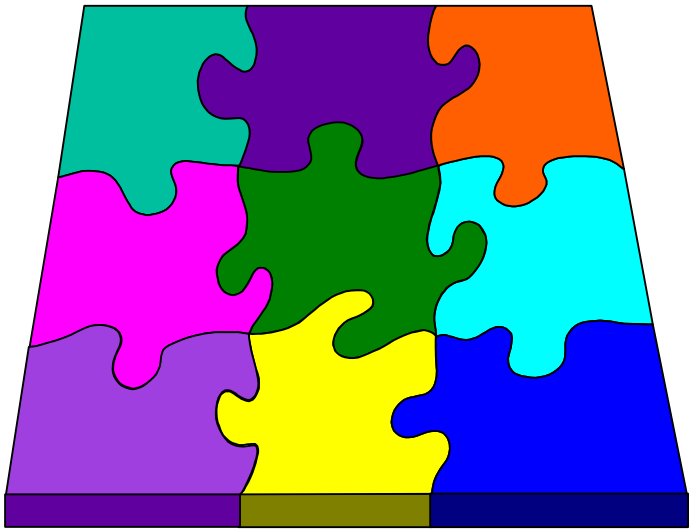
- *"Se você não atacar os riscos [do projeto] ativamente, então estes irão ativamente atacar você."* (Tom Gilb).
  - A maioria dos PDS que seguem o modelo iterativo e incremental aconselha que as partes mais arriscadas sejam consideradas inicialmente.



2.4 Utilização da UML no modelo iterativo e incremental

UML no modelo iterativo e incremental

- A UML é independente do processo de desenvolvimento.
  - Vários processos podem utilizar a UML para modelagem de um sistema OO.
- Os artefatos de software construídos através da UML evoluem à medida que o as iterações são realizadas.
  - A cada iteração, novos detalhes são adicionados a esses artefatos.
  - Além disso, a construção de um artefato fornece informações para adicionar detalhes a outros.



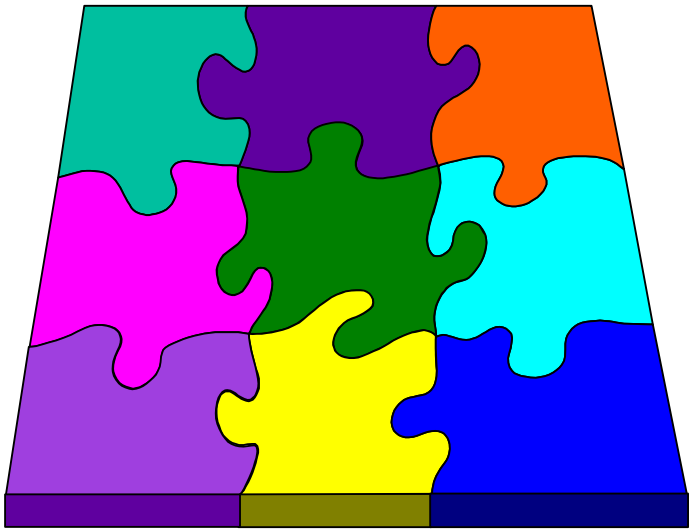
2.5 Prototipagem

Prototipagem

- A ***prototipagem*** é uma técnica aplicada quando:
  - há dificuldades no entendimento dos requisitos do sistema
  - há requisitos que precisam ser mais bem entendidos.
- A construção de ***protótipos*** utiliza ambientes com facilidades para a construção da interface gráfica.
- Procedimento geral da prototipagem:
  - Após o LR, um protótipo é construído para ser usado na *validação*.
  - Usuários fazem críticas...
  - O protótipo é então corrigido ou refinado
  - O processo de revisão e refinamento continua até que o protótipo seja aceito.
  - Após a aceitação, o protótipo é descartado ou utilizado como uma versão inicial do sistema.

Prototipagem

- Note que a prototipagem NÃO é um substituto à construção de modelos do sistema.
  - A prototipagem é uma técnica *complementar* à construção dos modelos do sistema.
  - Mesmo com o uso de protótipos, os modelos do sistema devem ser construídos.
  - Os erros detectados na validação do protótipo devem ser utilizados para modificar e refinar os modelos do sistema.



2.6 Ferramentas de suporte

Ferramentas de suporte

- O desenvolvimento de um software pode ser facilitado através do uso de ferramentas que auxiliam:
  - na construção de modelos,
  - na integração do trabalho de cada membro da equipe,
  - no gerenciamento do andamento do desenvolvimento, etc.

Ferramentas de suporte

- Há diversos sistemas de software que são utilizados para dar suporte ao desenvolvimento de outros sistemas.
- Um tipo bastante conhecido de ferramenta de suporte são as ***ferramentas CASE***.
  - CASE: *Computer Aided Software Engineering*
- Além das ferramentas CASE, outras ferramentas importantes são as que fornecem suporte ao ***gerenciamento***.
  - desenvolver cronogramas de tarefas,
  - definir alocações de verbas,
  - monitorar o progresso e os gastos,
  - gerar relatórios de gerenciamento, etc.

Ferramentas de suporte

- Criação e manutenção da consistência entre estes diagramas
- *Round-trip engineering*
- Depuração de código fonte
- Relatórios de testes
- Testes automáticos
- Gerenciamento de versões
- Verificação de desempenho
- Verificação de erros em tempo de execução
- Gerenciamento de mudanças nos requisitos
- Prototipagem