SSC0530- Introdução a Sistemas de Informação

Desenvolvimento de Sistemas de Informação

Profa. Ellen F. Barbosa

ICMC/USP

Os 7 Princípios Básicos de Desenvolvimento de Software

(by David Hooker)

Primeiro Princípio

A razão pelo qual tudo existe

- Tem que existir uma razão para se fazer software!!
  - Senão melhor não fazer!
  - Fazer software: **agregar valor para o usuário**.
  - É importante enxergar os reais requisitos do software!

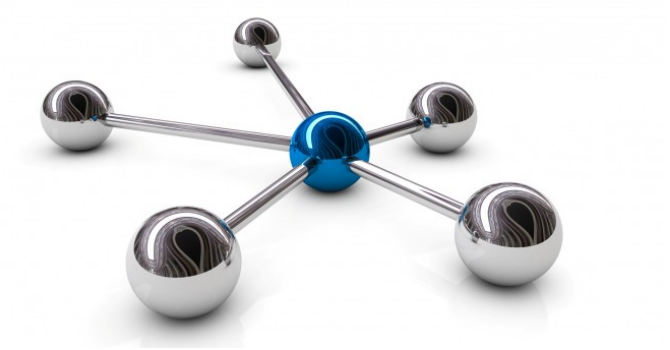
Segundo Princípio

Mantenha as coisas simples (KISS - Keep It Simple, Stupid!)

- **"Um projeto deve ser o mais simples possível, mas não mais simples que isso".**
- As soluções mais elegantes normalmente são simples.
- Fazer algo simples usualmente demanda mais tempo do que fazer de forma complexa.

Terceiro Princípio

Mantenha o estilo



- O projeto de um software deve seguir um único estilo (estilo de codificação, documentação, teste, um mesmo processo, ...).
- A combinação de diferentes estilos corretos pode levar a um software incorreto.
- Padrões e estilos devem ser estabelecidos no início e seguidos por todos.

Quarto Princípio

O que você produzir alguém vai consumir!

- Sempre especifique, projete e codifique algo pensando que **outros** vão ler.
- Sempre exija **qualidade** nos produtos que você consome e forneça qualidade nos produtos que você produz.
- Alguém pode ter que depurar o código que você escreve, e isso faz dele um usuário de seu código.

Quinto Princípio

Esteja aberto para o futuro!

- Sistemas de boa qualidade têm vida longa.
- Projete desde o início pensando na **manutenção**.

Sexto Princípio

Planeje com antecedência para reutilização

- Pense no problema geral, e não só no problema específico.
- Busque por soluções já existentes.

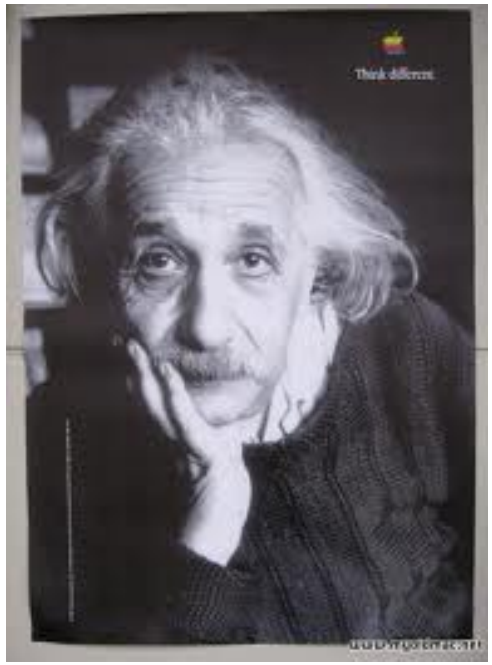
Sétimo Princípio

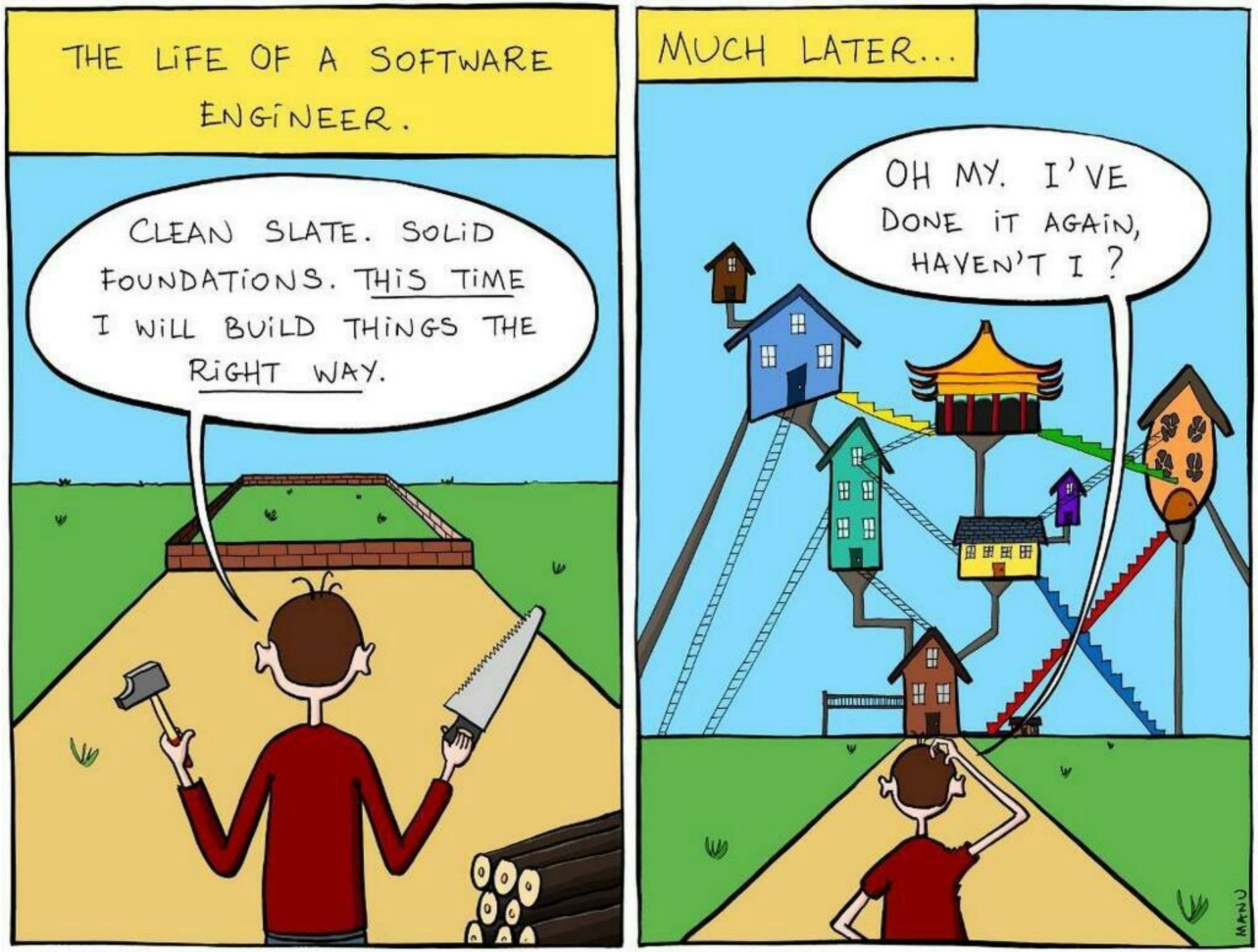
Pense !

"Plano é desnecessário, mas planejar é indispensável."

D. Eisenhower – General do exército americano

- Avalie alternativas.
- Detalhe os riscos.




THE LIFE OF A SOFTWARE ENGINEER.
CLEAN SLATE. SOLID FOUNDATIONS. THIS TIME I WILL BUILD THINGS THE RIGHT WAY.
MUCH LATER...
OH MY. I'VE DONE IT AGAIN, HAVEN'T I ?
MANU

Passos básicos para resolução de problemas e desenvolvimento de SI

1. **Entender o problema**
   a) definir o problema
   b) Identificar causas
   c) Identificar objetivos da solução
   d) Identificar requisitos de informação

2. **Desenvolver soluções alternativas**
   a) Identificar soluções alternativas

3. **Escolher a melhor solução**
   a) Avaliar as alternativas
   b) Escolher a melhor solução

4. **Implementar e testar a solução**
   a) Especificar o projeto
   b) Adquirir ou desenvolver o software
   c) Testar o sistema
   d) Preparar treinamento e documentação

Exemplo1 – Entender o problema

Preciso de um sistema que calcula o **mdc** entre dois números inteiros positivos diferentes de zero, usando o **algoritmo proposto por Euclides**.

Exemplo1 – Entender o problema

- O algoritmo de Euclides consiste em efetuar divisões sucessivas entre dois números até obter resto zero.
- O máximo divisor comum entre os dois números iniciais é o último resto diferente de zero obtido.

Exemplo1 – Entender o problema

Exemplo:

- **mdc de 10 e 15:**
- 15/10 = 1 resto 5
- 10/5 = 2 resto 0, logo o mdc é 5
- **mdc de 22 e 13:**
- 22/13 = 1 resto 8
- 13/8 = 1 resto 5
- 8/5 = 1 resto 3
- 5/3 = 1 resto 2
- 3/2 = 1 resto 1
- 2/1 = 2 resto 0, logo o mdc é 1

Exemplo 1 – Soluções alternativas

- Receber os valores de entrada para o cálculo ou gerar aleatoriamente?
- Usar vetores? Matrizes? Alocação dinâmica?
- Precisa armazenar os valores (arquivos)?
- Usar algoritmo recursivo?
  - Vantagens/desvantagens

Exemplo 2

Preciso de um sistema automatizado de caixa eletrônico de banco.

Exemplo 2 – Entender o problema

Preciso de um sistema automatizado de caixa eletrônico de banco.

O caixa deve permitir que o cliente, já cadastrado no banco realize saques, transferências, consultas e pagamentos.

Exemplo 2 – Entender o problema

Preciso de um sistema automatizado de caixa eletrônico de banco.

O caixa deve permitir que o cliente, já cadastrado no banco realize saques, transferências, consultas e pagamentos.

O cliente pode consultar o saldo ou tirar um extrato de sua conta.

Exemplo 2 – Entender o problema

Preciso de um sistema automatizado de caixa eletrônico de banco.

O caixa deve permitir que o cliente, já cadastrado no banco realize saques, transferências, consultas e pagamentos.

O cliente pode consultar o saldo ou tirar um extrato de sua conta.

Cliente pode ter mais de uma conta de tipos diferentes.

Cliente precisa fazer login no sistema para qualquer funcionalidade.

Exemplo 2 – Entender o problema

Preciso de um sistema automatizado de caixa eletrônico de banco.

O caixa deve permitir que o cliente, já cadastrado no banco realize saques, transferências, consultas e pagamentos.

O cliente pode consultar o saldo ou tirar um extrato de sua conta.

Cliente pode ter mais de uma conta de tipos diferentes

Cliente precisa fazer login no sistema para qualquer funcionalidade

Que tipos de relatórios são fornecidos?

Como o sistema interage com outros sistemas?

Quais funções existem do lado da administração do caixa?

Exemplo 2 – Entender o problema

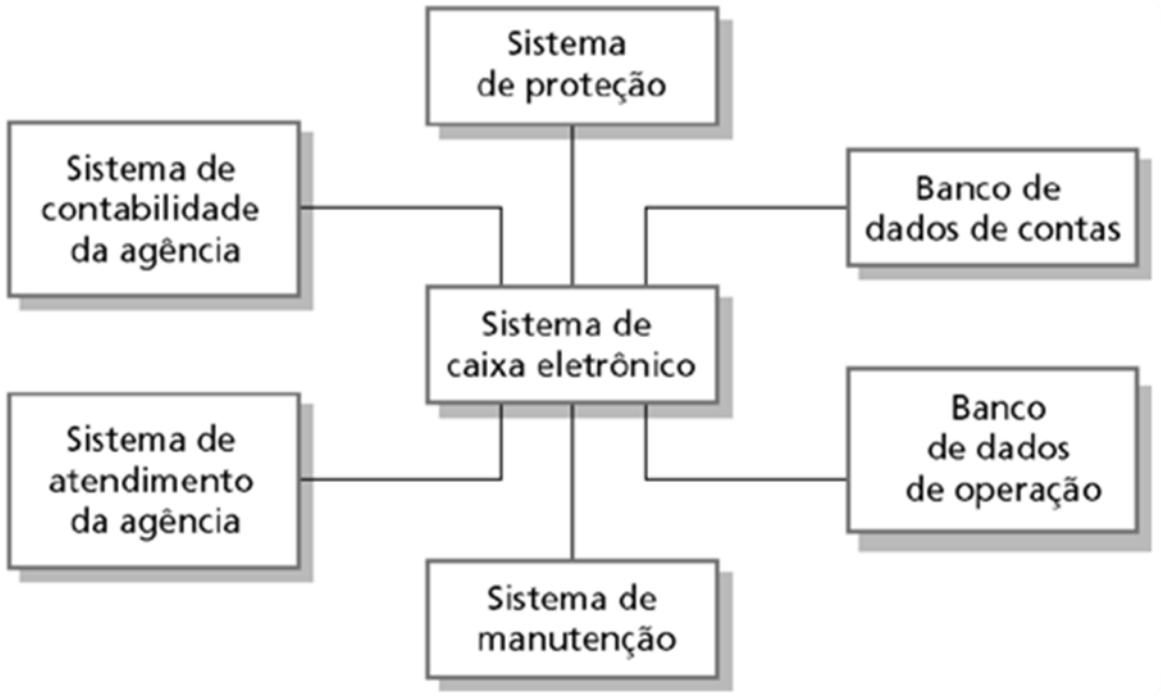
Preciso de um sistema automatizado de caixa eletrônico de banco.

O caixa deve permitir que o cliente, já cadastrado no banco realize saques, transferências, consultas e pagamentos.

O cliente pode consultar o saldo ou tirar um extrato de sua conta.

Figura 8.1

Contexto de um sistema de caixa eletrônico.



Exemplo 2 – Soluções alternativas

- Tipos de hardware possíveis.
- Divisão do sistema em partes que representam as funcionalidades
  - Onde fica o login? Em todas as funcionalidades?
- Segurança do sistema.
- Linguagem de desenvolvimento.
- Interface do sistema?

Passos para o desenvolvimento

- Após escolhido a melhor solução: **implementar a solução!**

- **Testar a solução**
  - O que testar?
  - Como testar?
  - Teste de unidade, teste de sistema, teste de aceitação.

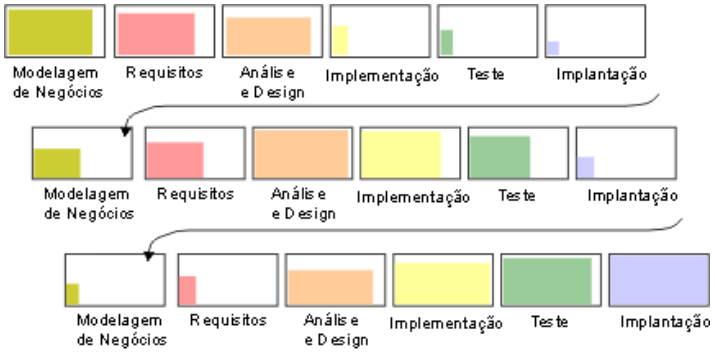
Princípio básico de sucesso!!!

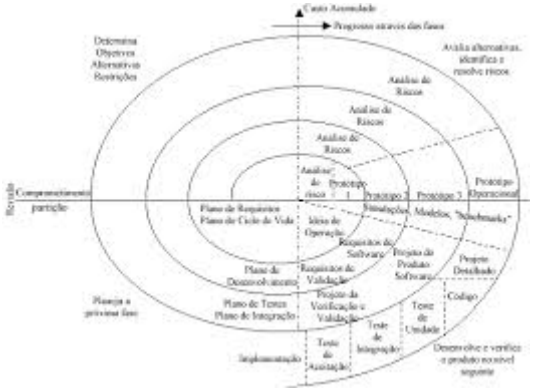
Pense na solução (ou em várias) antes de implementar uma solução!

- Teste sua solução: funciona para todos os casos?
- É factível (tempo x custo)?
- Existe tecnologia para implementá-la?

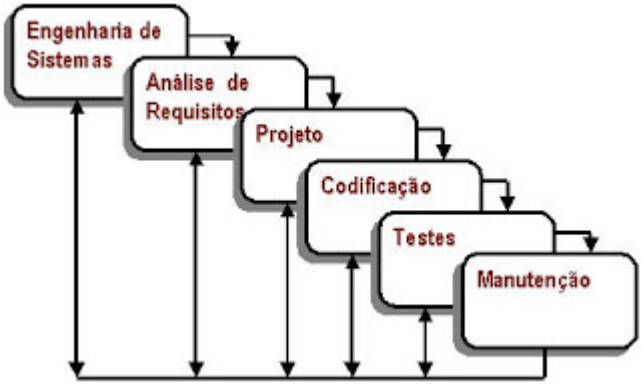
Exemplos de itens a serem especificados em um sistema

- Saída
- Entrada
- Interface de usuário
- Banco de dados
- Processamento (quais funções)
- Procedimentos manuais
- Segurança
- Mudanças organizacionais
- ...

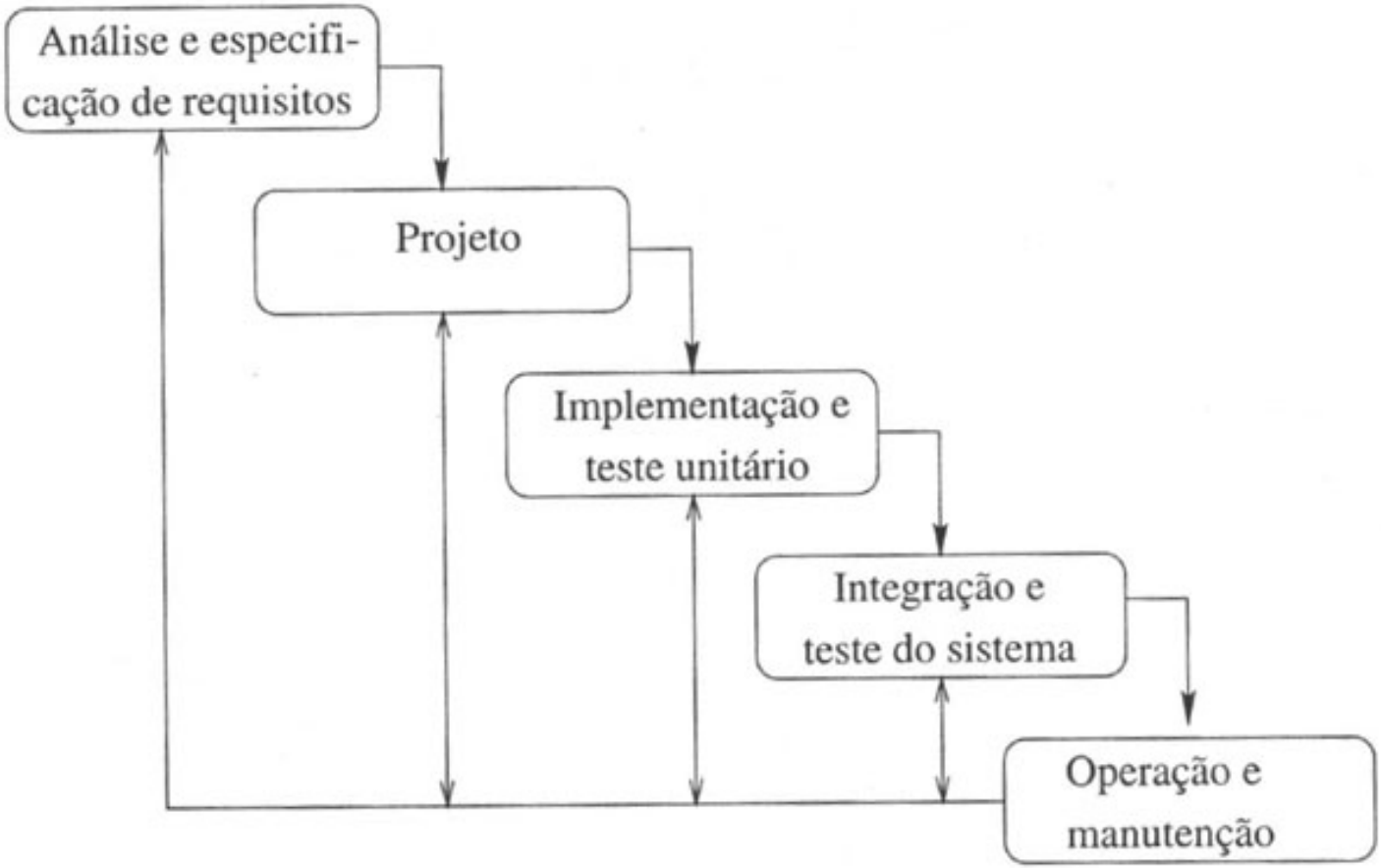




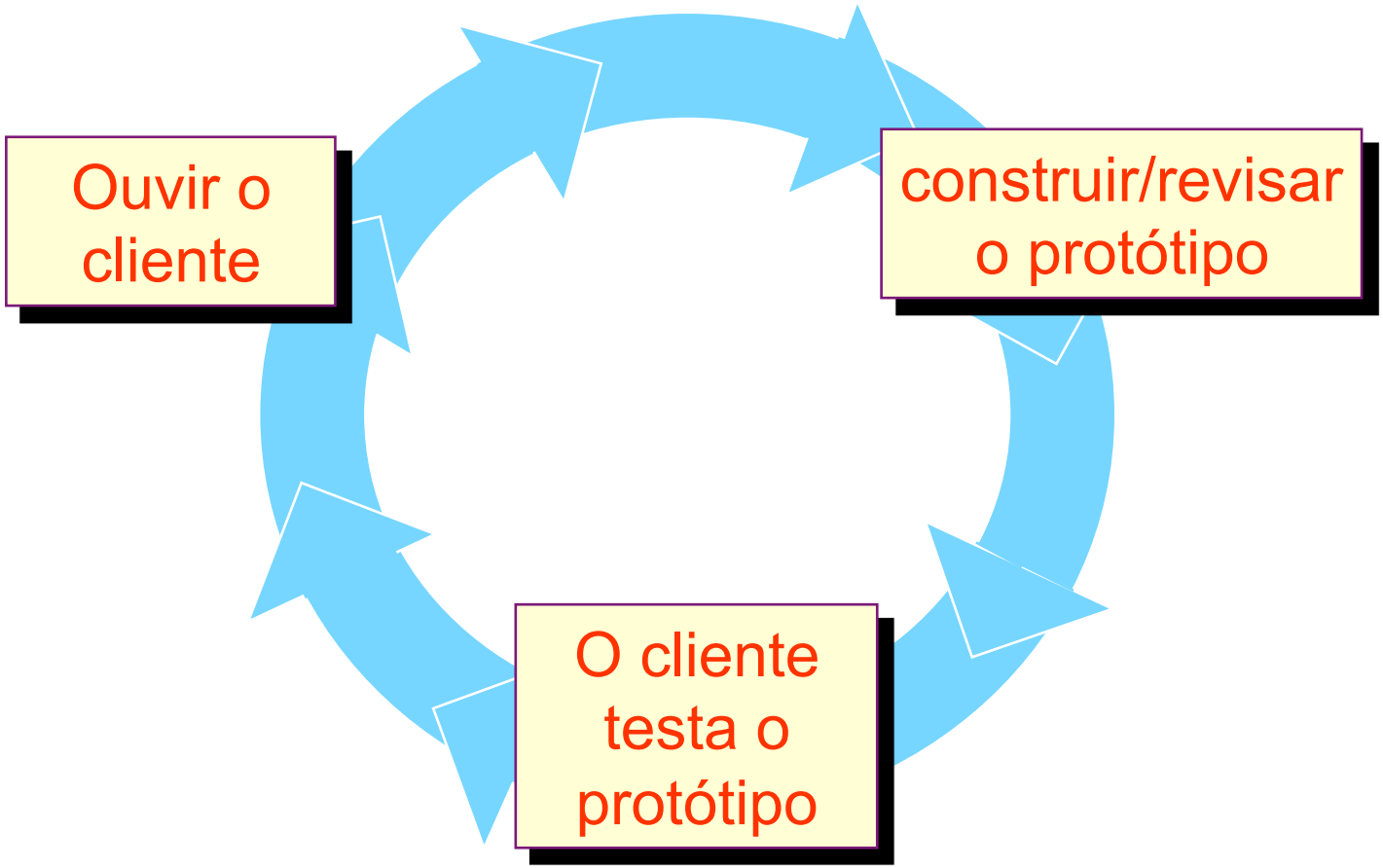
Como desenvolver o sistema?



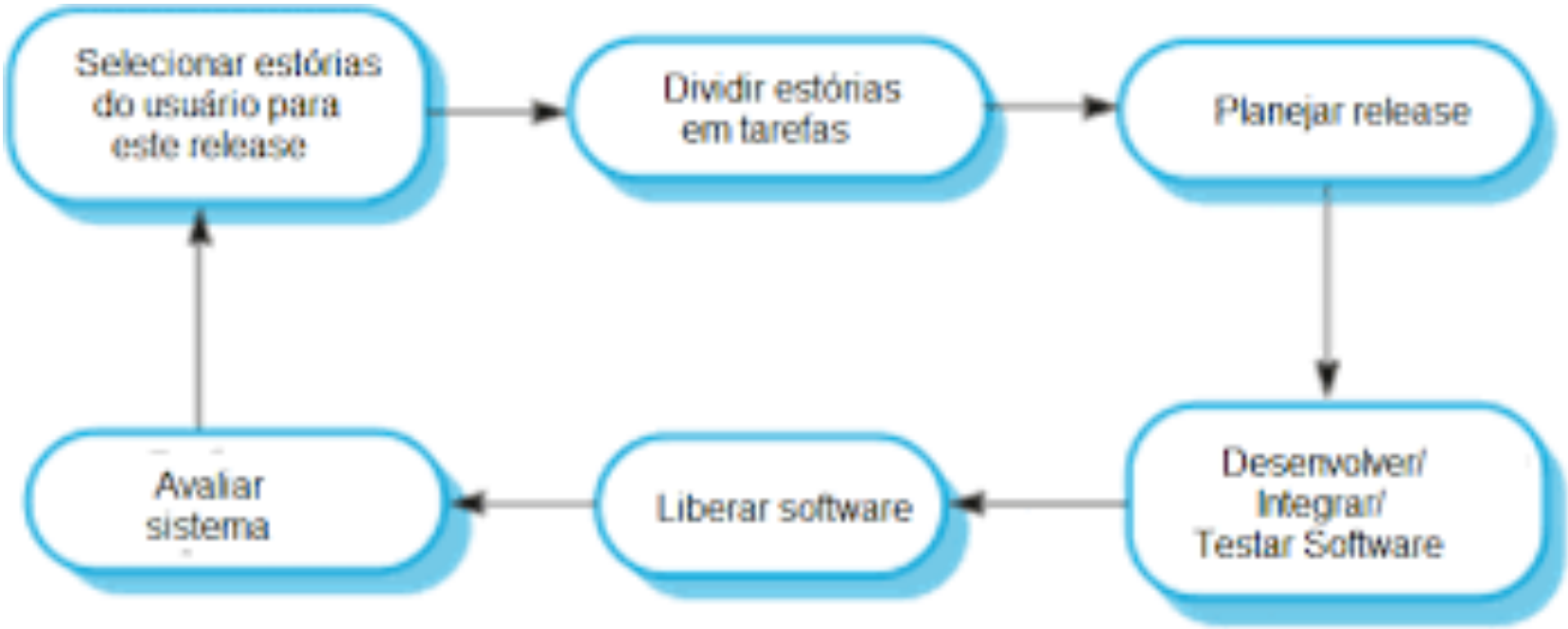
Desenvolvimento tradicional



Prototipação



Desenvolvimento ágil



Gerência de projetos

- **Definição de projeto**

É um empreendimento temporário com o objetivo de criar um produto ou serviço único (PMBOK).

Gerência de projetos

- É a aplicação de conhecimentos, habilidades e técnicas para projetar atividades que visem atingir os objetivos do projeto.
- Relacionado com:
  - Demandas:
    - Escopo
    - Tempo
    - Risco
    - Qualidade
  - Interesses e necessidades (*stakeholders*)
  - Identificação de objetivos

Gerência de projetos

Gerência de projetos

- Planejar o projeto
  - Definir objetivo, escopo, tempo, custo.
  - Riscos associados.

- Monitorar o projeto
  - Acompanhar a evolução.
  - Aplicar plano de ação no caso de riscos.

Conclusões

- **Qualidade do software** deve ser preocupação constante.
- **Planejar** antes de realizar.
- Testar seu sistema de maneira adequada -> **qualidade**!
- Objetivo principal sempre: **Satisfação do Cliente!**

Dúvidas??

- **Rua Carlos Botelho, 1869**

Dúvidas??

- Rua Carlos Botelho, 1869

- Arquivei
  - Quarta, 14/06, 13:30 hrs