TESTE DE SISTEMAS

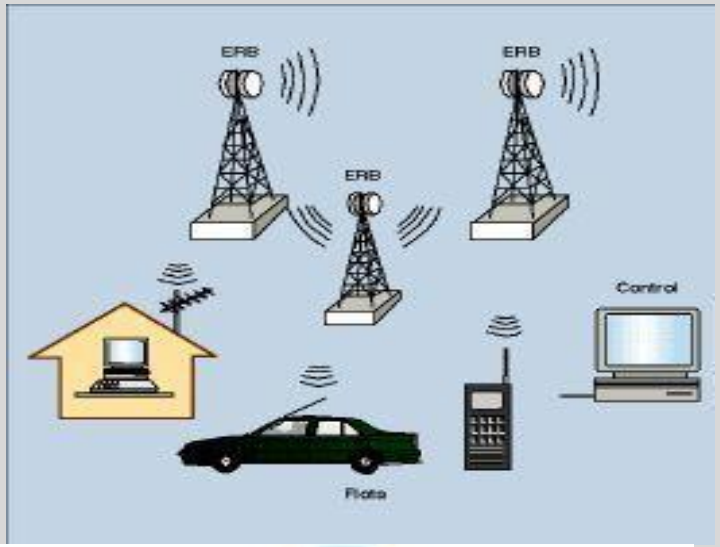
Professor Alann Perini

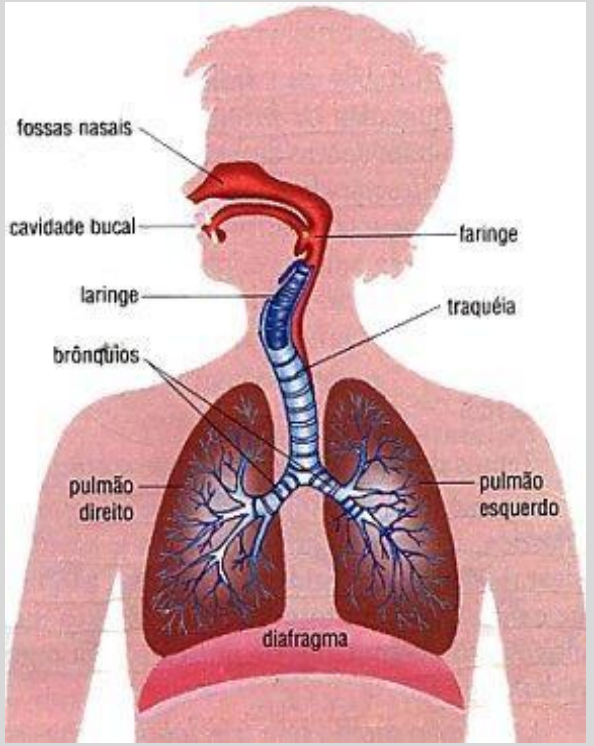
CONCEITOS RELACIONADOS A ANÁLISE DE SISTEMAS,

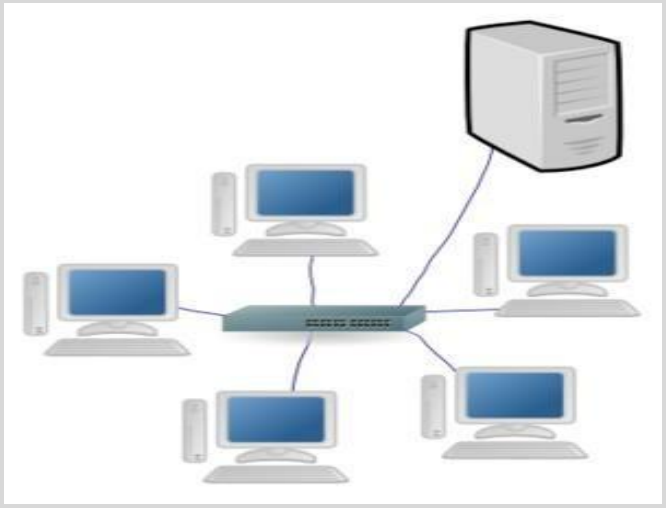
UML,

Teste de Sistemas

Tipos de Sistemas







- Combinação de elementos relacionados entre si

A natureza dos sistemas...

- Vimos que existem muitos sistemas;
- Os sistemas artificiais (de informação) objetivam:
- Fornecer, pesquisar, controlar, prover e analisar informações, tomada de decisões;
- Estão associados à empresas/organizações e consideram dois aspectos:
  - Componentes;
  - Nível de decisão;

Análise Estruturada do Sistema

Necessidade do usuário/ Levantamento de Requisitos

Concepção do Sistema

Análise de viabilidades

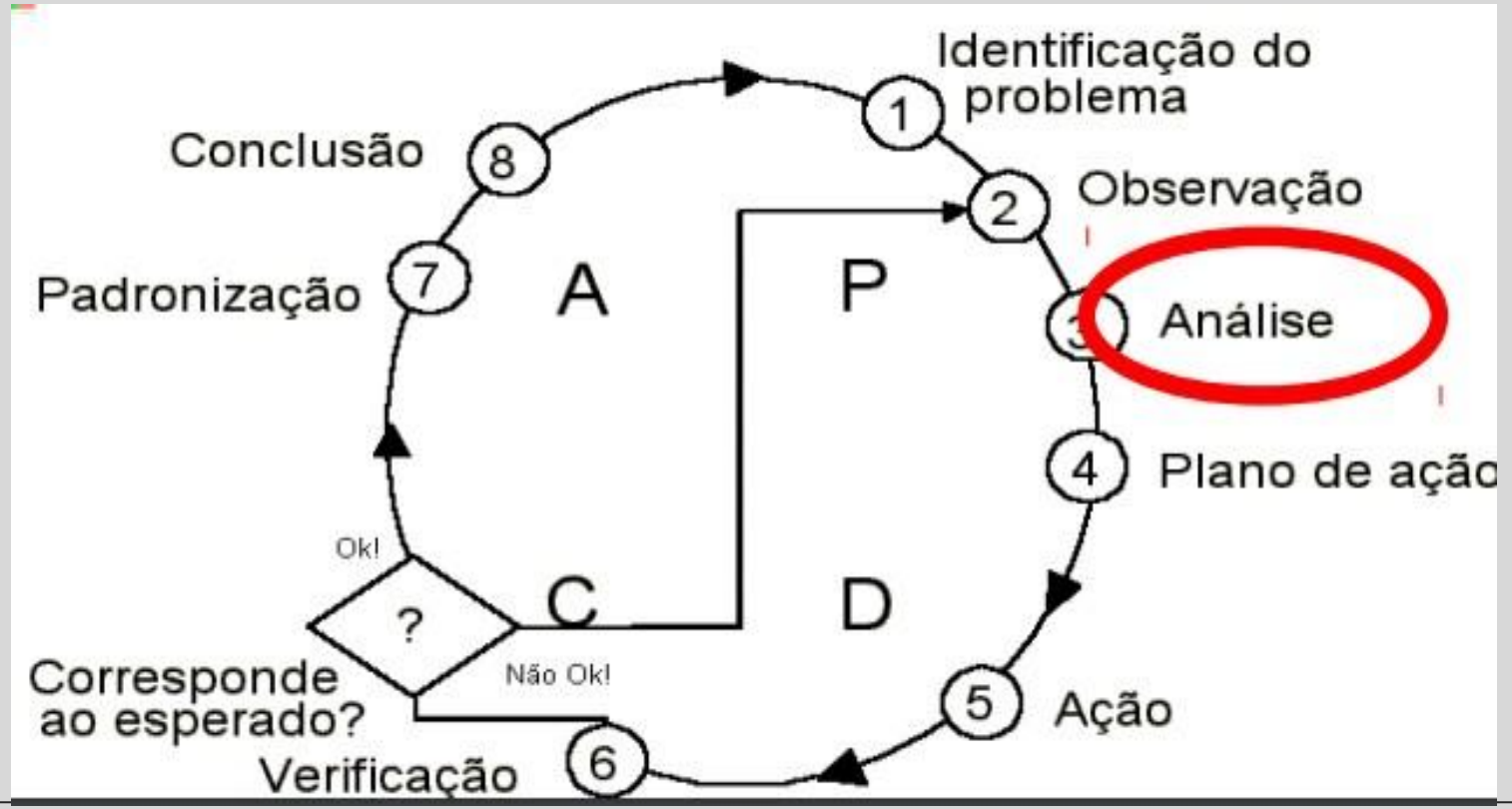
Projeto lógico

Projeto físico

Implantação/Testes

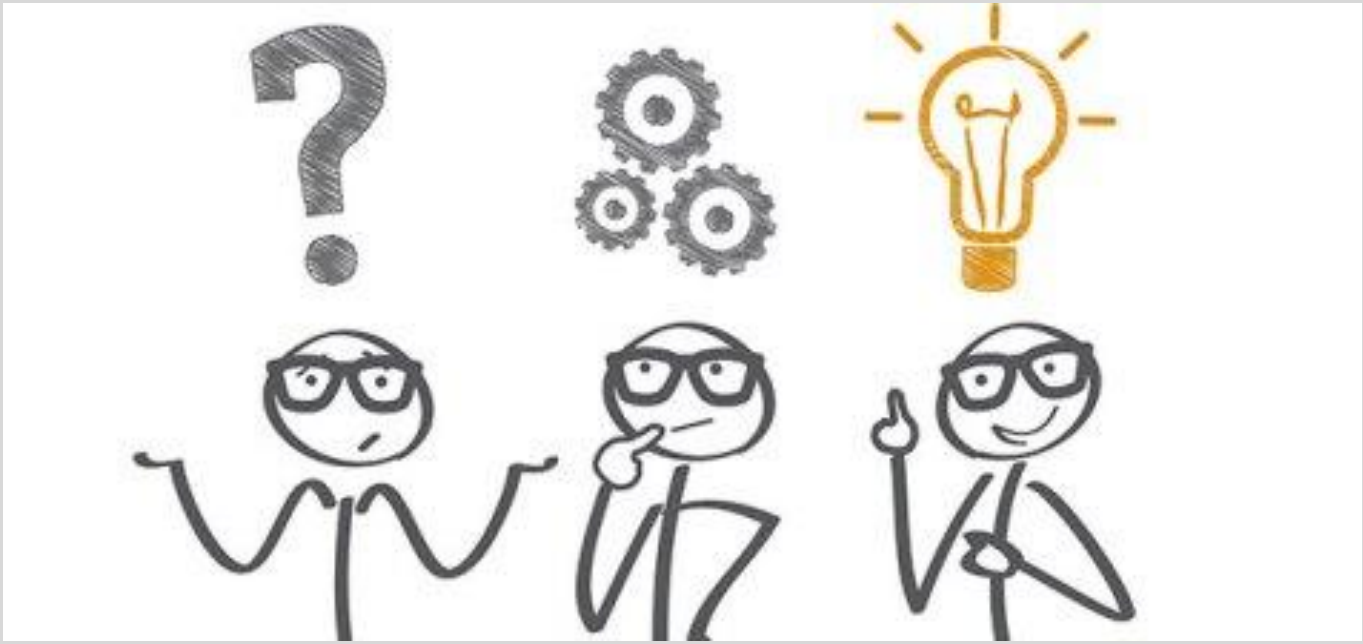
Manutenção

- Para se construir estes sistemas necessita-se primeiro entendê-los;
- Mas o que os sistemas tem em comum? O que devemos entender?
  - ✔ Problemas !!!
- A resolução de um problema envolve:



A análise de sistemas...

- Consiste nos métodos e técnicas de avaliação e especificação da solução de problemas, para implementação em algum meio que a suporte, utilizando mecanismos apropriados;



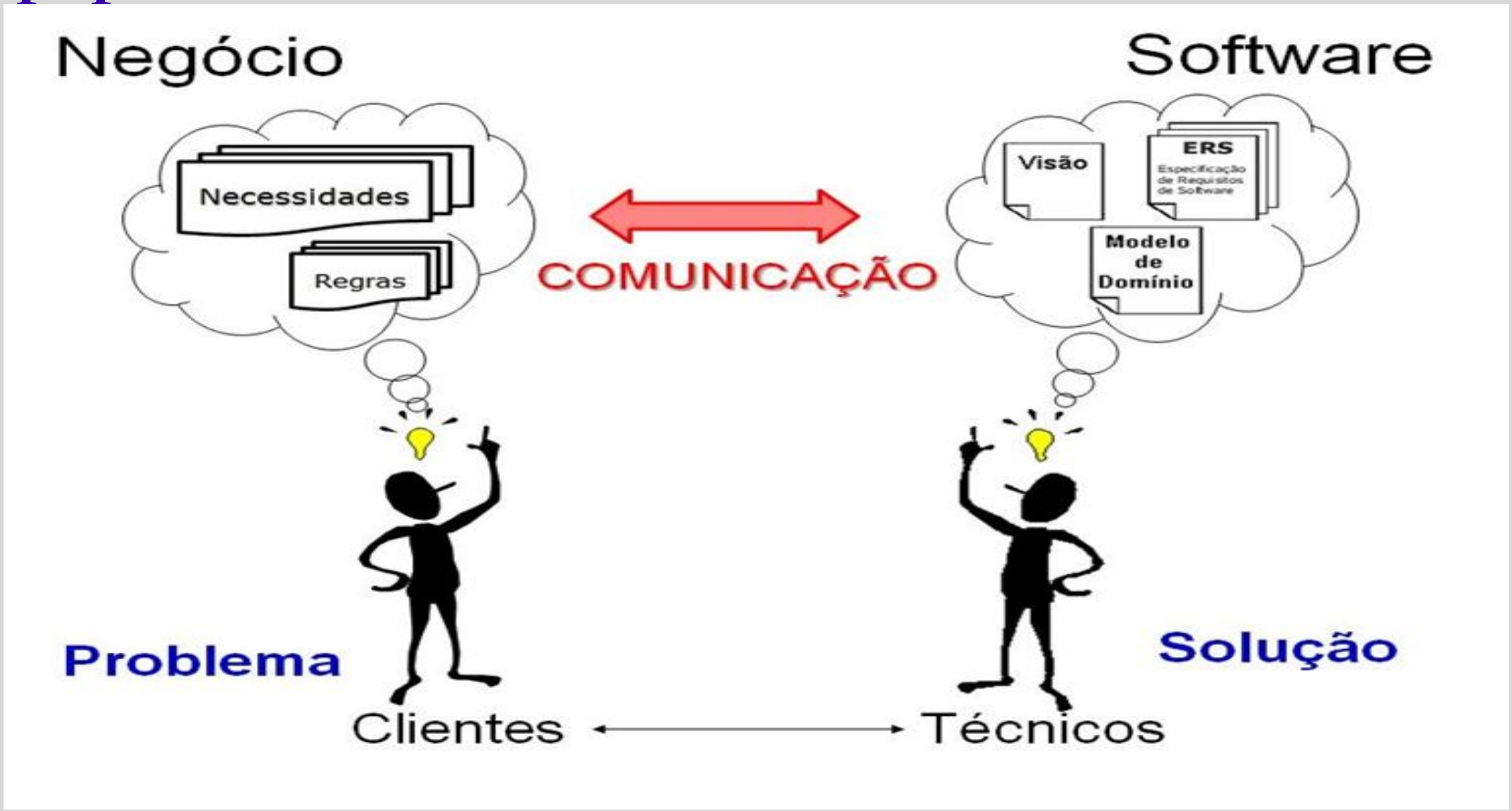
O papel do analista de sistemas...

- Elo entre usuários e computadores;
- Deve entender e avaliar as necessidades e expectativas de cada usuário, a fim de que estas sejam organizadas e especificadas seguindo uma formalidade técnica;
- Tem de ser capaz de lidar, ao mesmo tempo, com:
  - *grupos de usuários;*
  - *outros profissionais de informática;*
  - *corpo administrativo (gerentes/diretores)*
  - *Cada um trará formações, pontos de vistas, vivências, experiências e maturidade totalmente distintas;*

O papel do Analista de Sistemas

Iniciativa
Criatividade
Concentração
Persuasão
Comunicativo
Autoconfiança
Simplicidade
Ação Conciliadora
Clareza de Raciocínio
Espírito de Grupo
Percepção
Sensibilidade
Persistência
Flexibilidade
Determinação

O papel do Analista de Sistemas



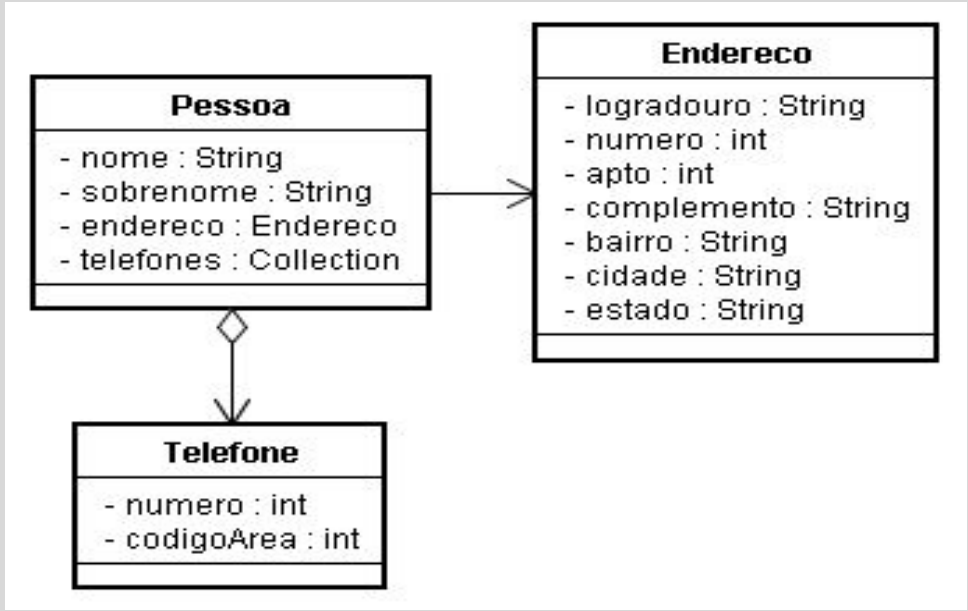
Métodos de projeto de Sistemas

- Método = Caminho (etapas);
- Há basicamente oito grandes métodos:

1. Metodologia Ágil. ...
2. Metodologia Cachoeira. ...
3. Metodologia DevOps. ...
4. Metodologia Aplicação rápida. ...
5. Metodologia Espiral. ...
6. Metodologia Protótipo. ...
7. Metodologia Programação extrema (XP) ...
8. Metodologia **Desenvolvimento** orientado a recursos.

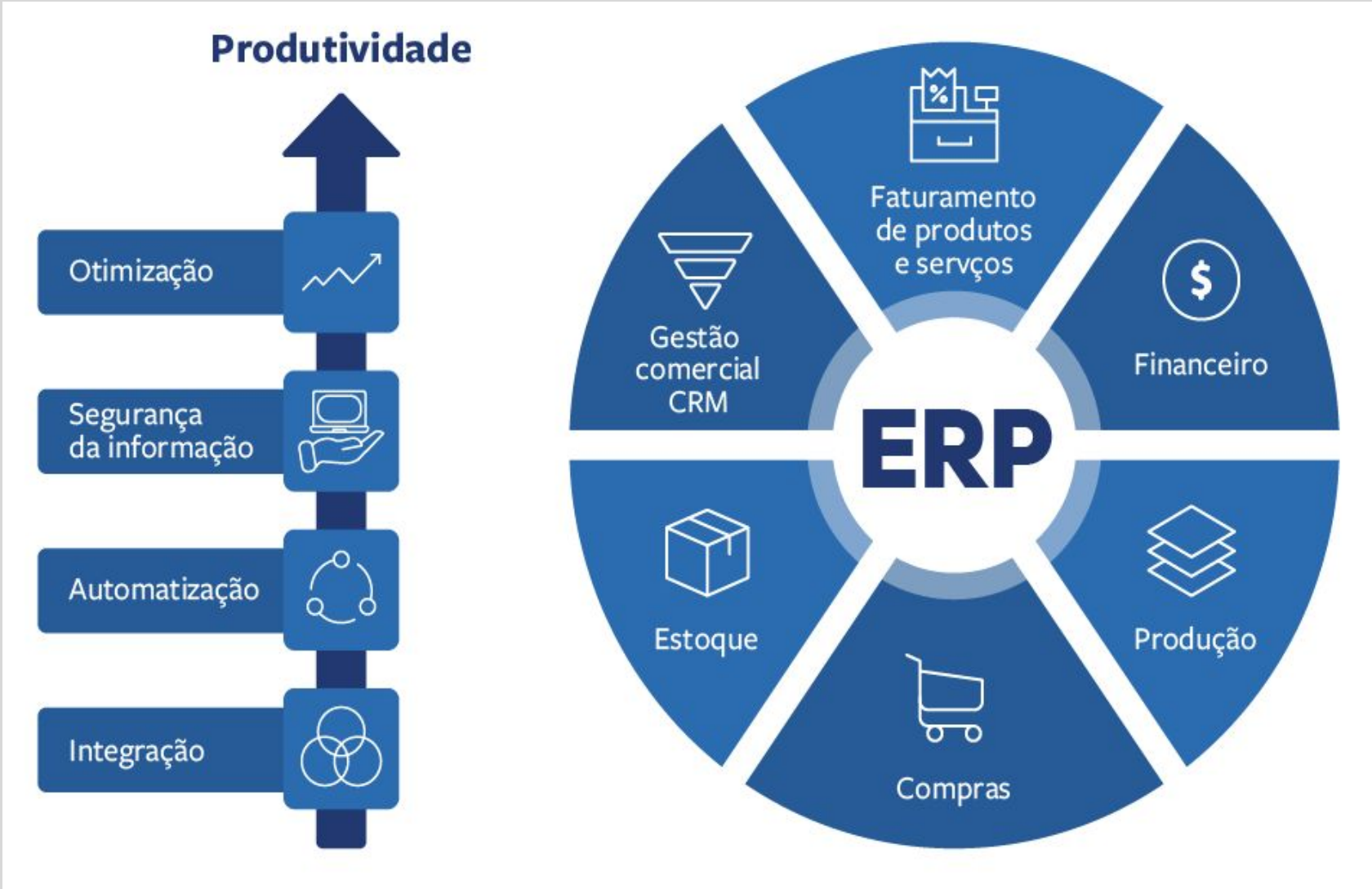
Ferramentas

- São usadas para ajudar a trilhar o caminho;
- São instrumentos que ajudam o analista durante o desenvolvimento de um sistema;
- Exemplos:
  - ✔ Entrevistas;
  - ✔ Simbologias e Diagramas;
  - ✔ Quadros de acompanhamento e evolução;



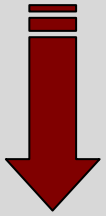
Sistemas ERP (Enterprise Resource Planning)

- São compostos por uma base de dados única e por módulos que suportam diversas atividades das empresas;



Definição de Requisitos

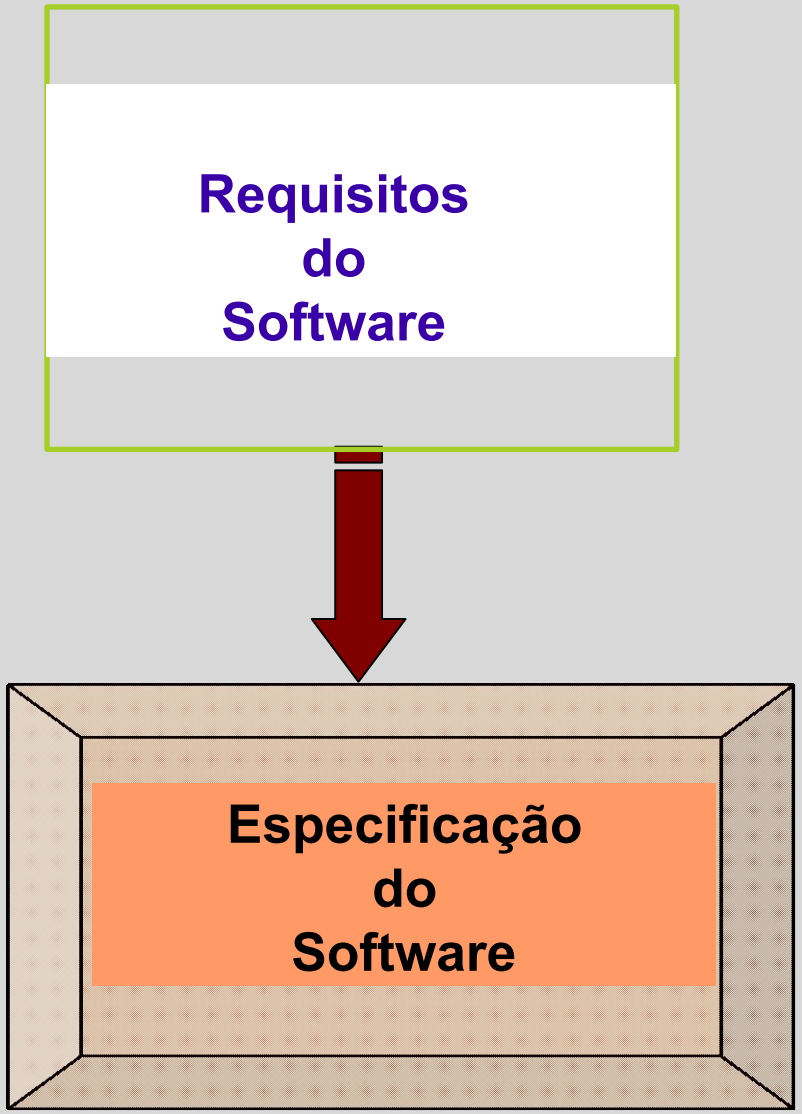
Desejos Intenções
Procedimentos
Dados



Requisitos
do
Software

- Identificar
  - desejos, intenções, procedimentos atuais e dados;
- Organizá-los de forma coerente
- Definir de uma forma geral
  - o que será tratado pelo software
  - interface com o que fica de fora do software

Análise do Software



- Entendimento e Representação
  - Domínio do problema
  - Conceitos
  - Funcionalidades
  - Casos de uso

- Baseado nos fatores críticos de sucesso do software

Diagramas da UML

- **Visão Externa**
  - Diagrama de Casos de Uso
- **Visão Estrutural (Estática)**
  - Diagrama de Classes
- **Visão Comportamental (Dinâmica)**
  - Diagrama de Estado
  - Diagrama de Atividade
- **Visão de Interação**
  - Diagrama de Sequência
  - Diagrama de Colaboração
- **Visão da Arquitetura (Implementação)**
  - Diagrama de Componentes
  - Diagrama de Implantação

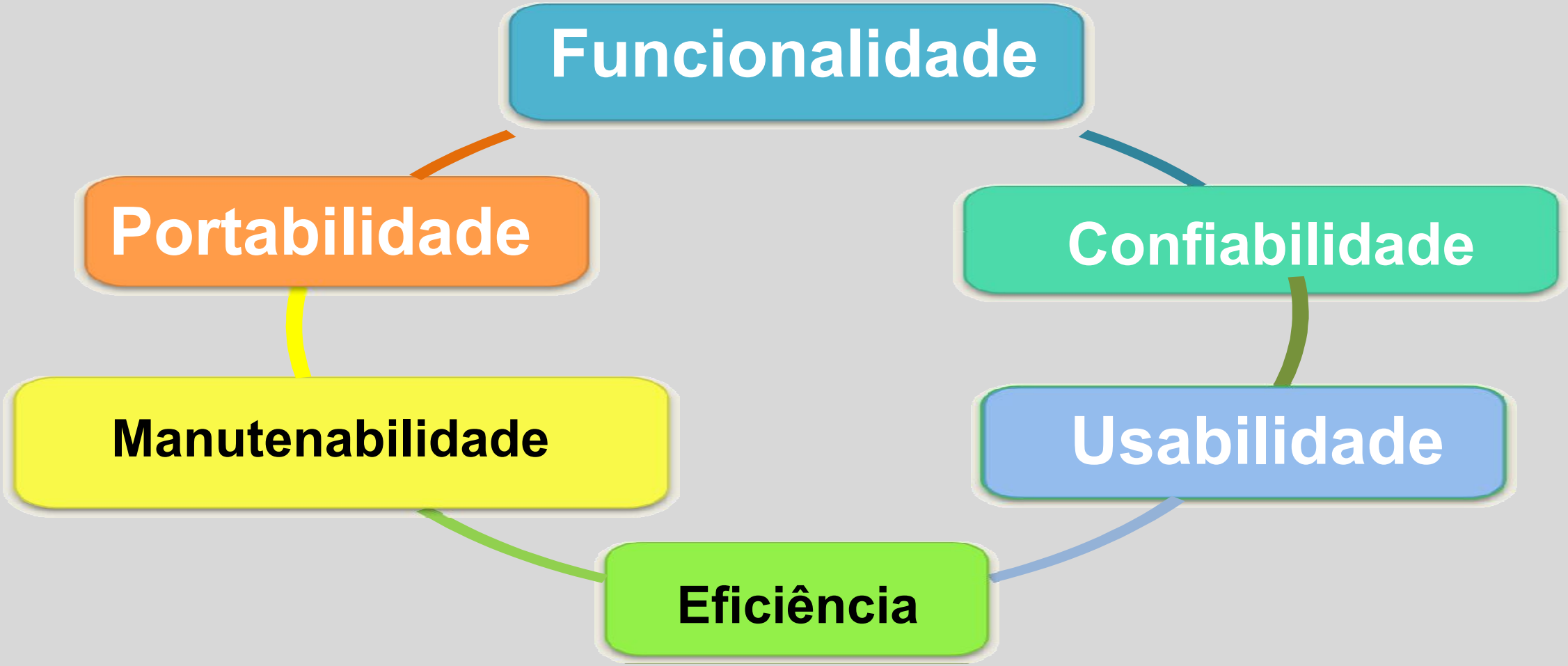
Testes de Software

TESTE DE SOFTWARE

O teste de software mostra as falhas do sistema antes que o desenvolvimento seja concluído. A partir desse teste, é possível assegurar que as funcionalidades solicitadas estejam presentes e de acordo com o esperado.

Quando as falhas são corrigidas nas fases finais do projeto, o custo é maior do que se fossem solucionadas no início. Por isso, muitas empresas, profissionais e equipes preferem fazer um desenvolvimento orientado aos testes.

A Norma ISO 9126 define as seguintes características para qualidade:



O QUE É TESTE DE SOFTWARE

Os testes são realizados com a intenção de descobrir erros e defeitos em um sistema. [Myres, 2004]

Os testes de software podem ser usados para mostrar a presença de defeitos, mas nunca para mostrar a ausência deles. [Dijkstra, 1972]

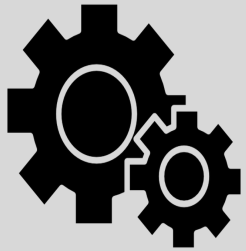
O QUE É TESTE DE SOFTWARE

Os testes de software servem para medir a confiabilidade de um sistema: à medida que poucos defeitos são encontrados em um determinado tempo, o software é considerado mais confiável.

Processo de desenvolvimento de software

Requisitos do Sistema

Modelo do Sistema



Projeto do Sistema

```
def add5(x):
    return x+5

def dotwrite(ast):
    nodename = getNodename()
    label=symbol.sym_name.get(int(ast[0]),ast[0])
    print '     %s [label="%s' % (nodename, label),
    if isinstance(ast[1], str):
        if ast[1].strip():
            print '= %s"];' % ast[1]
        else:
            print '"]'
    else:
        print '"];'
        children = []
        for  in n, childenumerate(ast[1:]):
            children.append(dotwrite(child))
        print ,'     %s -> {' % nodename
        for  in :namechildren
            print '%s' % name,
```

Implementação do Sistema

Teste do Sistema

Processo de desenvolvimento de software

```
def add5(x):
    return x+5

def dotwrite(ast):
    nodename = getNodename()
    label=symbol.sym_name.get(int(ast[0]),ast[0])
    print '    %s [label="%s' % (nodename, label),
    if isinstance(ast[1], str):
        if ast[1].strip():
            print '= %s"];' % ast[1]
        else:
            print '"]'
    else:
        print '"];'
        children = []
        for  in n, childenumerate(ast[1:]):
            children.append(dotwrite(child))
        print ,'    %s -> {' % nodename
        for  in :namechildren
            print '%s' % name,
```

Implementação do Sistema

Teste do Sistema

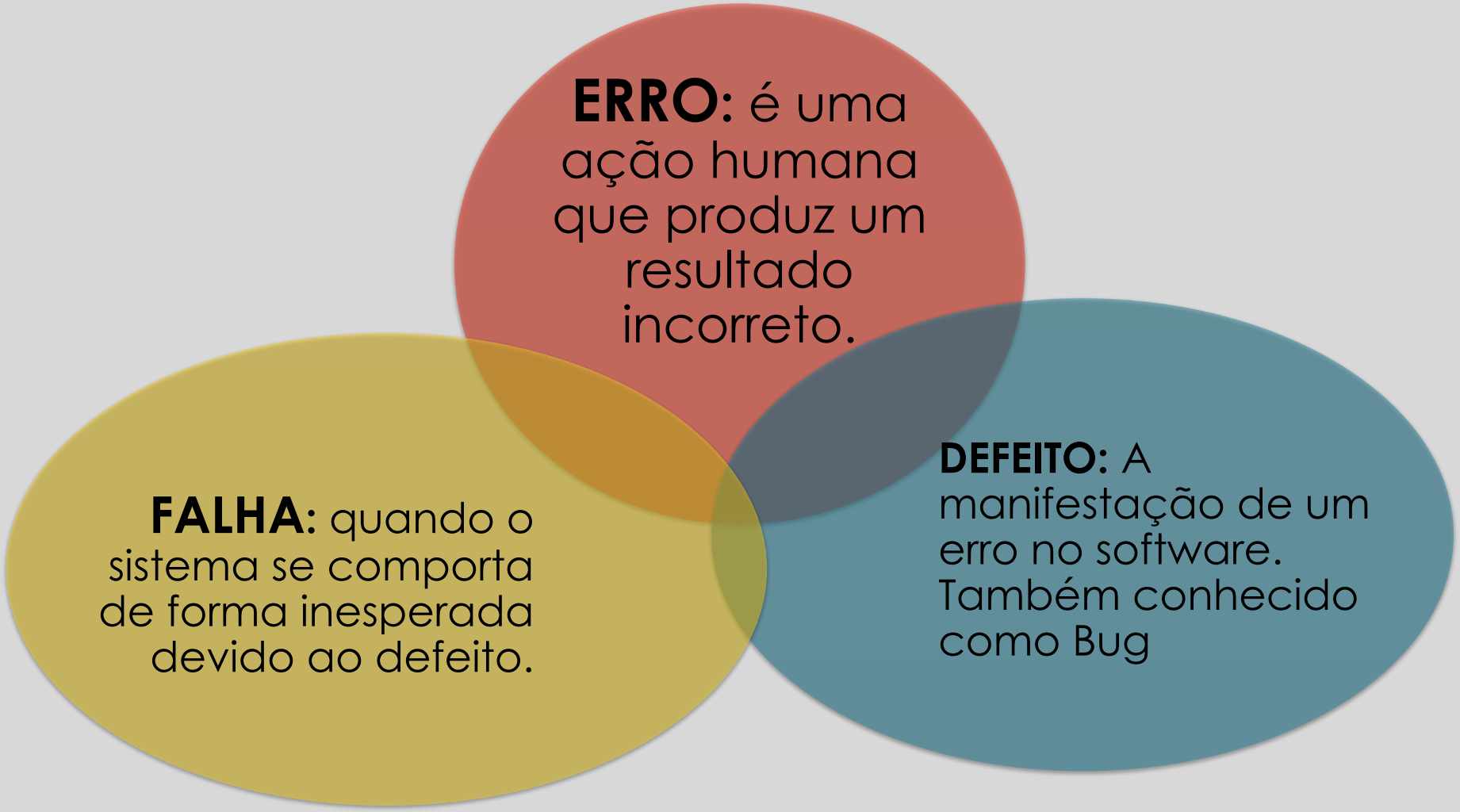
Objetivos do teste

- ✔ Demostrar que o software está livre de defeitos ?
- ✔ Encontrar a existência de falhas!
- ✔ Os testes podem mostrar apenas a presença de defeitos, mas não a sua ausência.

PORQUE TESTAR É NECESSÁRIO?

- ✔ Para assegurar que as necessidades dos usuários estejam sendo atendidas.
- ✔ Porque é bem provável que o software possua defeitos.
- ✔ Desenvolvedor já alocado para outro projeto teria que resolver muitos bugs de projetos anteriores em produção.
- ✔ Porque falhas de software custam muito dinheiro.
- ✔ Para gerar uma avaliação da qualidade do software.

ERRO, DEFEITO E FALHA



ERRO, DEFEITO E FALHA

Uma pessoa comete um **ERRO**...

...que pode causar uma **FALHA** na operação.

...que cria um **DEFEITO** no software...

Conceitos Básicos de Teste

Artefatos de Teste → todo o conjunto de documentação gerado pelo processo de teste de software.

Caso de Teste → é composto por um conjunto de entradas, por passos de execução e um resultado esperado

Roteiro de Teste → É composto por um conjunto de casos de teste definidos para uma determinada especificação.

Conceitos Básicos de Teste

Requisitos → regras de negócio do sistema

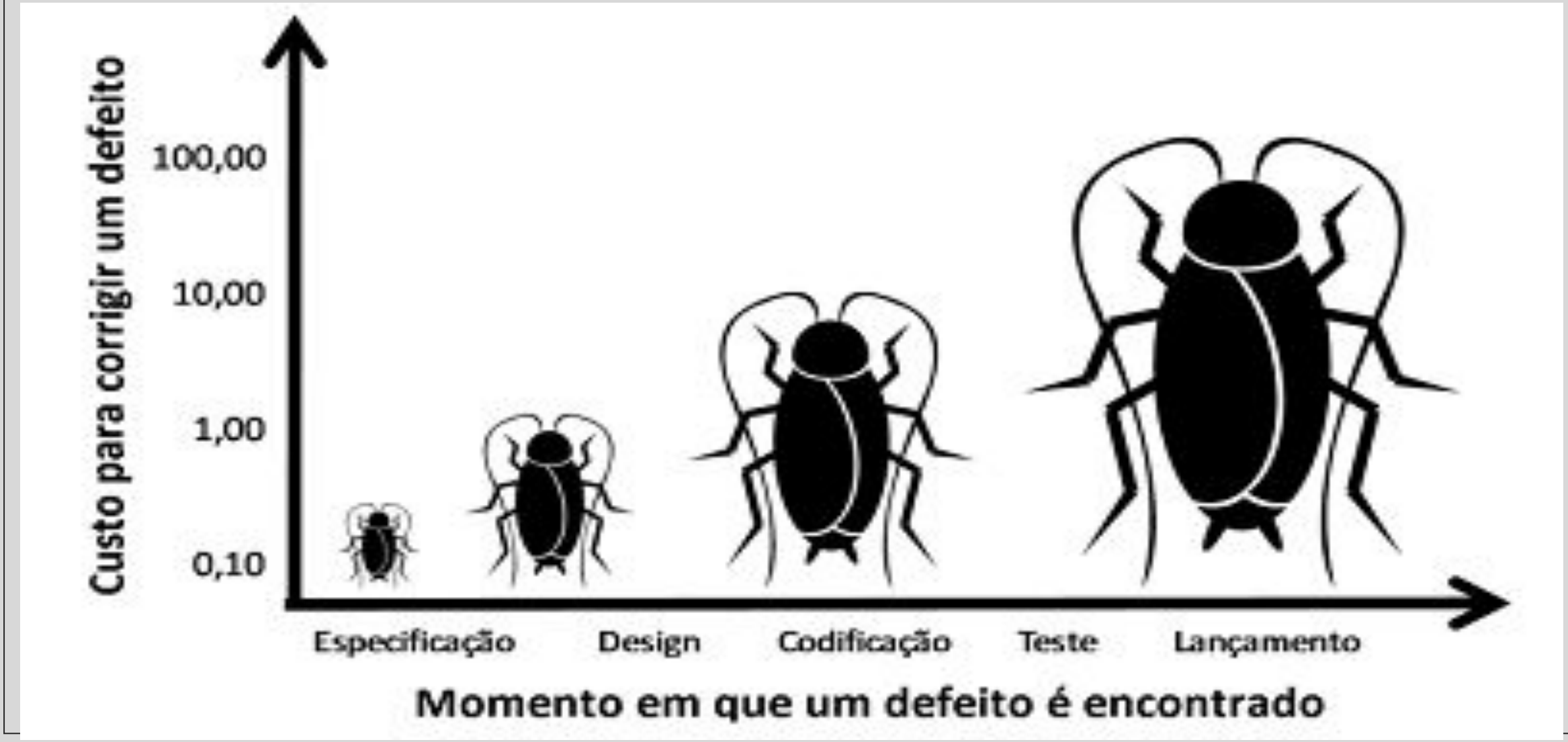
Testar → descobrir falhas através da execução do sistema.

Bug → é um defeito encontrado no sistema em execução.

Confiabilidade do Software

Confiabilidade do Software é a probabilidade que o software não causará uma falha no sistema por um tempo especificado, sob condições determinadas

Confiabilidade do Software



O CUSTO DE UM DEFEITO

O **custo** da correção de um defeito tende a ser cada vez **maior** quanto **mais tarde** ele for descoberto. [Myres, 2004]

O CUSTO DE UM DEFEITO

Defeitos, por definição, são imperfeições, deformidades, não funcionamento de um sistema ou mecanismo, falha. Eles são reflexos da não-conformidade.
Não-conformidades são o descumprimento de requisitos especificados.

Os defeitos custam dinheiro e podem ser vistos como o preço da não-conformidade. Em outras palavras, é o preço que decidimos pagar por não termos prevenido a ocorrência de defeitos no processo.

DESASTRES CAUSADOS POR ERROS EM SOFTWARES

Queda de um Airbus A320 em testes de aproximação

Aconteceu em 26 de junho de 1988: A queda do voo 296 da Air France com 136 passageiros. - Um voo de exibição do Airbus A320

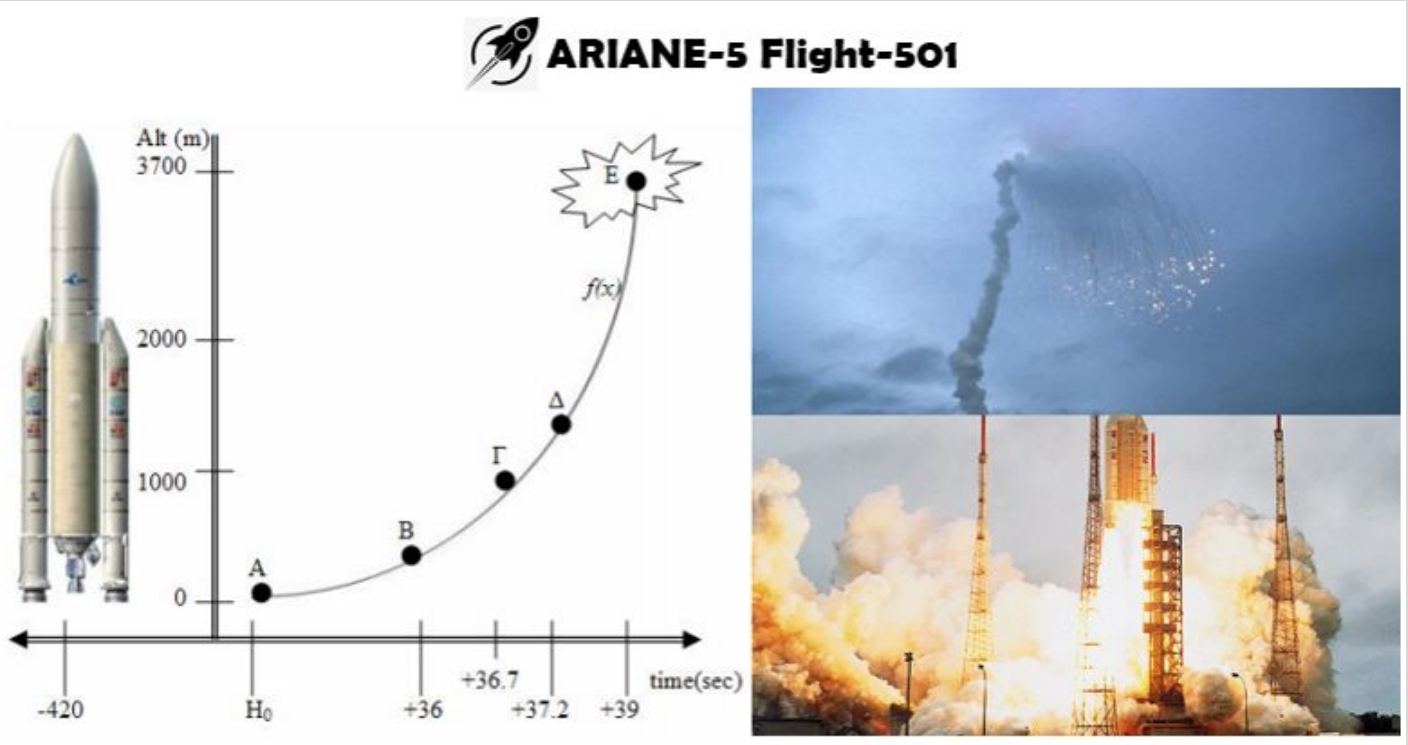
Fonte: https://youtu.be/c4cCVKkR28U

http://desastresaereosnews.blogspot.com/2021/06/aconteceu-em-26-de-junho-de-1988-queda.html

DESASTRES CAUSADOS POR ERROS EM SOFTWARES

- Assim como o Titanic “inafundável”, o A320 “intransponível” rapidamente encontrou seu iceberg proverbial: a confiança insuprível do ego humano.

O acidente com o Ariane 5 no vôo inaugural Um erro simples que custou $500 Milhões de dólares

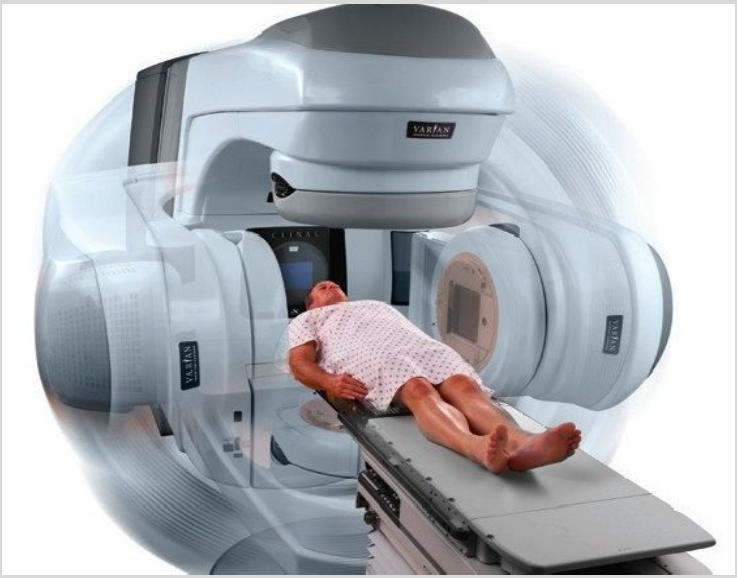


Motivo da falha: Erro de software no cálculo da velocidade horizontal do foguete. A variável que armazenava este valor tinha 64 bits (floating point) e foi erroneamente modificada para 16 bits (signed integer).

https://www.linkedin.com/pulse/ariane-5-flight-501-erro-de-eng-software-ou-sistemas-walter-pinotti/?originalSubdomain=pt

Fonte:https://youtu.be/72KES9VcnV0

Desastres causados por erros em softwares



No ano de 2000 um erro de cálculo no sistema de radioterapia, utilizado para controlar a emissão de radiação em tratamentos de câncer, *matou 8 pessoas e causou queimaduras graves em outras 20.*

MITOS SOBRE OS TESTES

O testador é inimigo do desenvolvedor

Os testadores devem ser os desenvolvedores menos qualificados.

O sistema está pronto quando o desenvolvedor termina de codificar.

Um programador consegue testar eficientemente o próprio código

TIPOS DE TESTE DE SOFTWARE

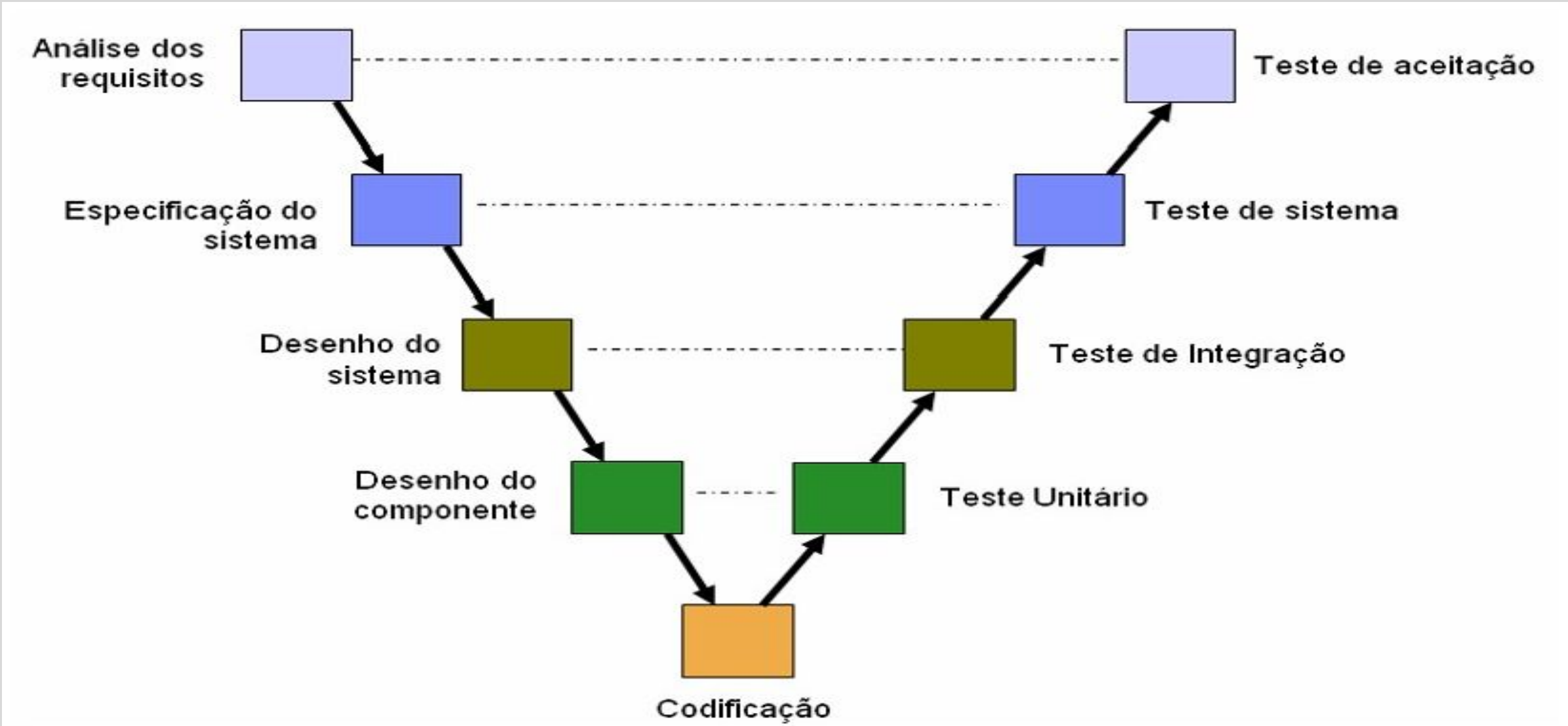
Testes de Caixa-Branca (Estrutural)

- Testes de Unidade
- Teste de Integração

Testes de Caixa-Preta (Funcional)

- Testes Funcionais
- Testes de Aceitação
- Testes Exploratórios

NÍVEIS DE TESTE DE SOFTWARE



Atributos	Nível dos Testes			
	Testes Unitários	**Testes de Integração**	**Testes de Sistema**	**Testes de Aceitação**
Escopo	*Unidades*	*Conjunto de unidades agrupadas*	*Sistema todo*	*Sistema todo*
Equipe	*Desenvolvedores*	*Desenvolvedores e Analistas de Sistema*	*Analista de Testes e Testadores*	*Analista de Testes, Testadores e Usuários*
Origem dos dados	*Criação manual*	*Criação manual*	*Criação automática / dados reais*	*Dados reais*
Volume dos dados	*Pequeno*	*Pequeno*	*Grande*	*Grande*
Interfaces	*Não existem*	*Não existem*	*Simuladas / Reais*	*Reais*
Ambientes	*Desenvolvimento*	*Desenvolvimento*	*Testes*	*Testes / Produção*

Atividade Avaliativa

A entrega deve ser feita em PDF até o final da aula de hoje!

Realize uma pesquisa dos tópicos e subtópicos abaixo contendo conceitos exemplos e modelos, quanto mais informação melhor!

Testes de Caixa-Preta (Funcional)

- Testes Funcionais
- Testes de Aceitação
- Testes Exploratórios

Testes de Caixa-Cinza

- Testes de Regressão

Testes de Caixa-Branca (Estrutural)

- Testes de Unidade
- Teste de Integração