ânima EDUCAÇÃO

UNISUL
UNIVERSIDADE DO SUL DE SANTA CATARINA

MODELAGEM DE SOFTWARE

Prof. Edson Lessa
Prof. Richard Henrique de Souza
Prof. Richard Schmitz
Prof. Ricardo Ribeiro Assink
Prof. Saulo Popov Zambiasi

Engenharia de Software

O que é?
Motivadores
O que é um software de qualidade?
O que são processos de desenvolvimento?
O que são ferramentas CASE?
O que é UML?

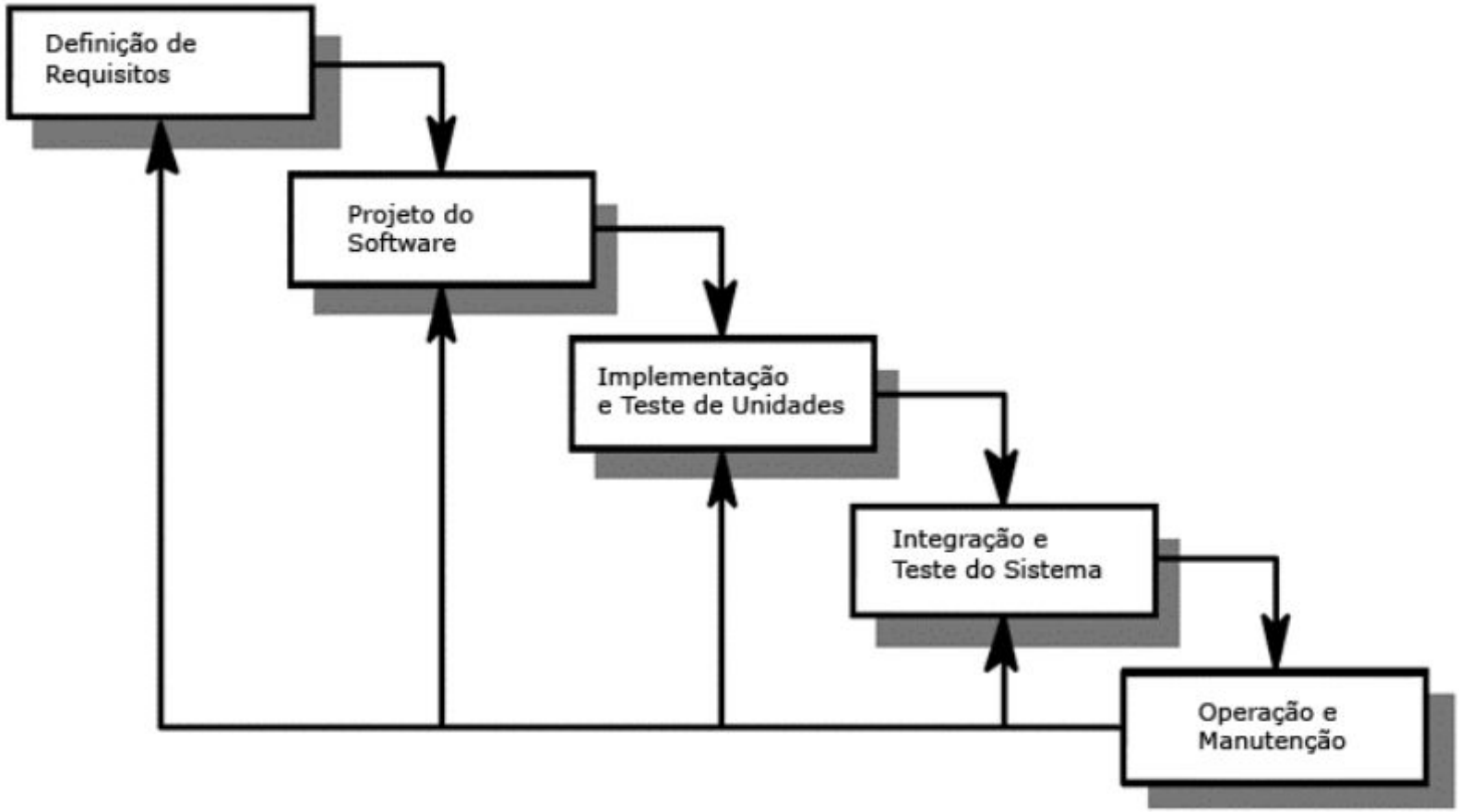
Engenharia de software

"Se preocupa com o projeto, implementação, instalação e operação de sistemas que incluem hardware, software e pessoas". (Sommerville)

Sistemas requerem abordagem multidisciplinar e devem ser projetados para durarem muitos anos em um ambiente dinâmico.

Podemos identificar três fases nos paradigmas de desenvolvimento.

1. **Definição**: Determina viabilidade, requisitos do software, especifica e projeta o sistema.
2. **Desenvolvimento**: Implementação, integração e instalação.
3. **Operação**: manutenção, correção e evolução.

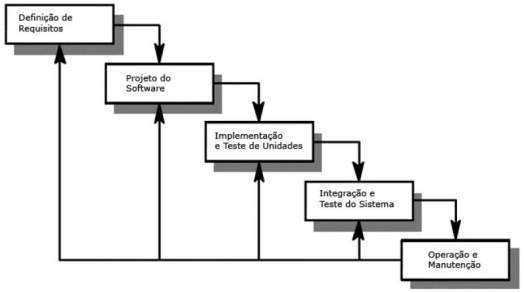


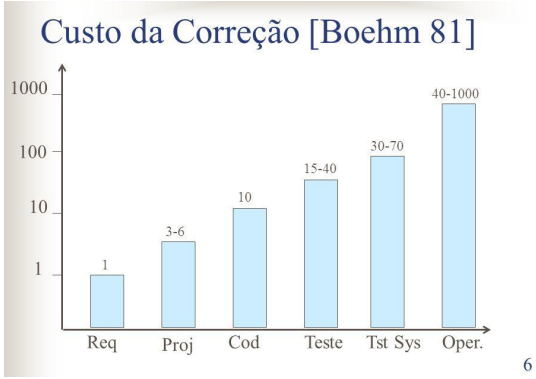
Modelo de Desenvolvimento de SW

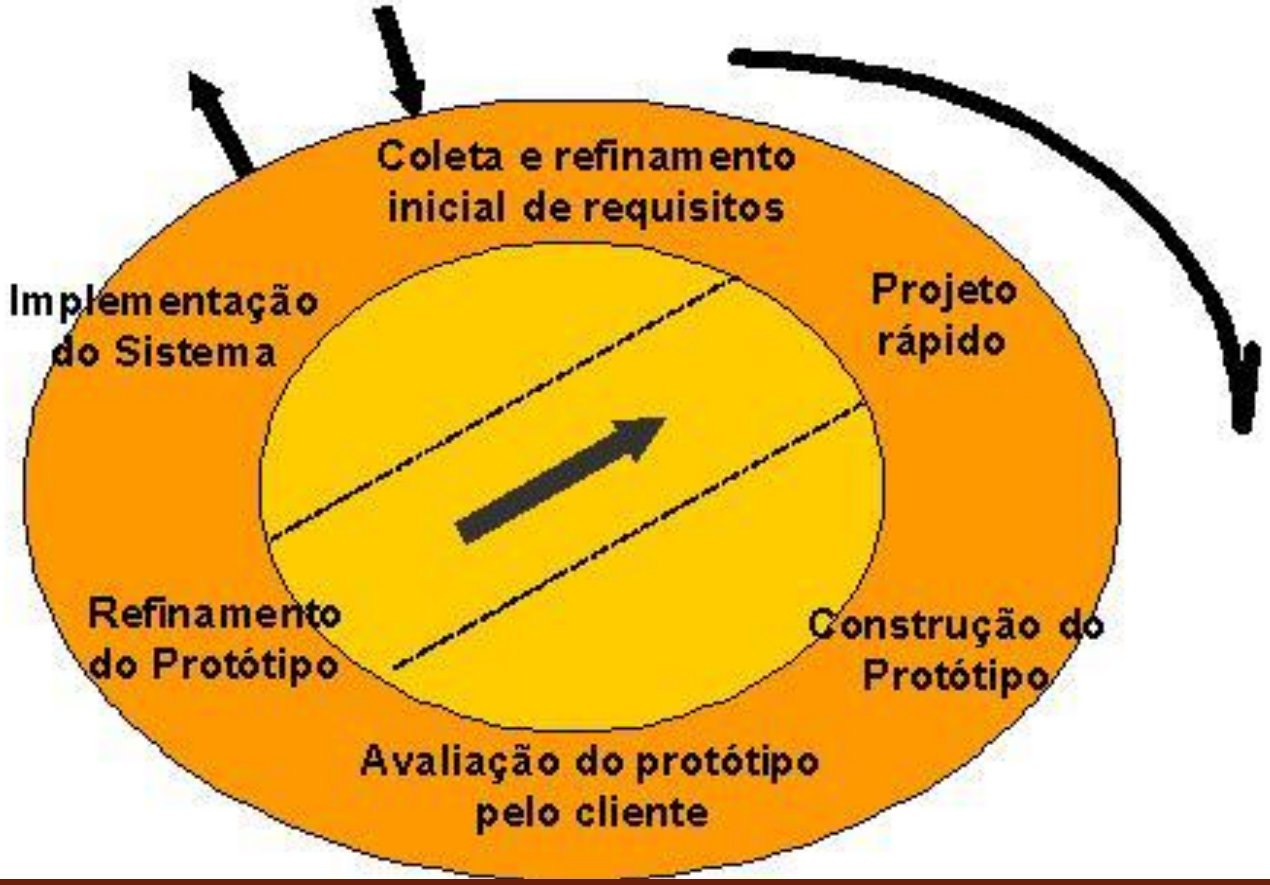
Modelo em cascata

- Demora na entrega do produto.
- Acúmulo de riscos.
- Inflexível a mudanças nos requisites.

Porém, muito usado em outros modelos. Interessante para requisitos estáveis.

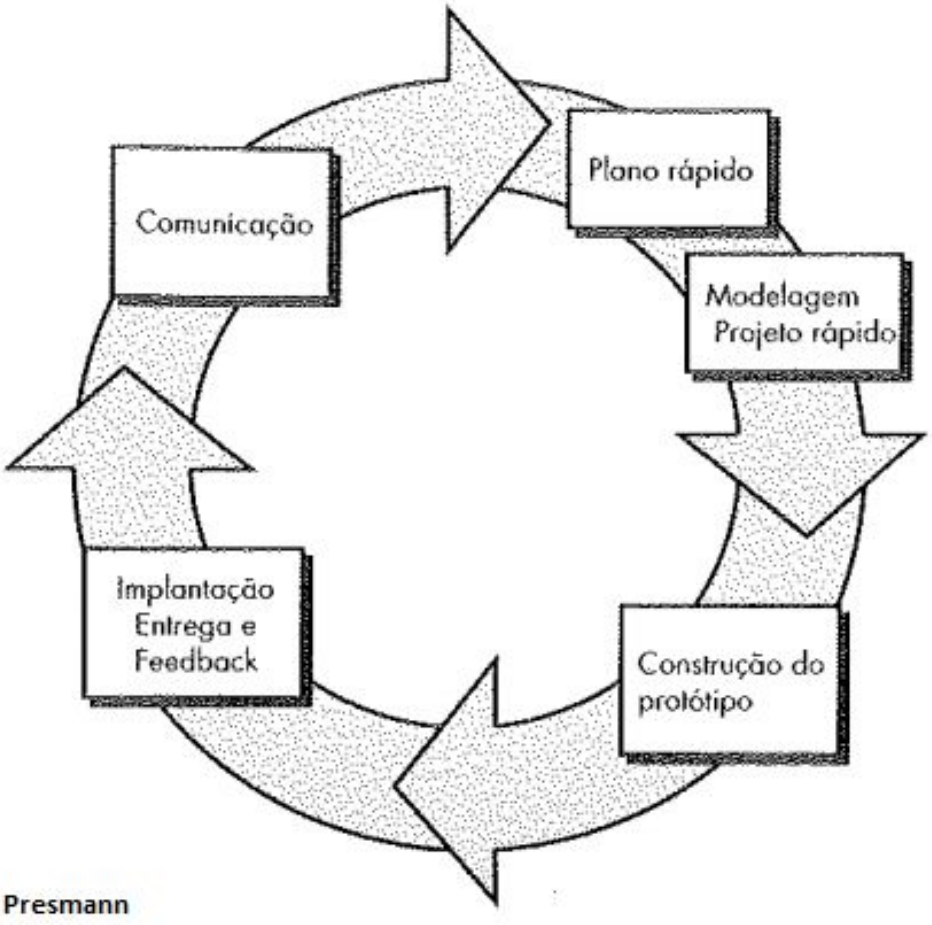






Modelo de Desenvolvimento de SW

Prototipagem



Presmann

Evolucionária

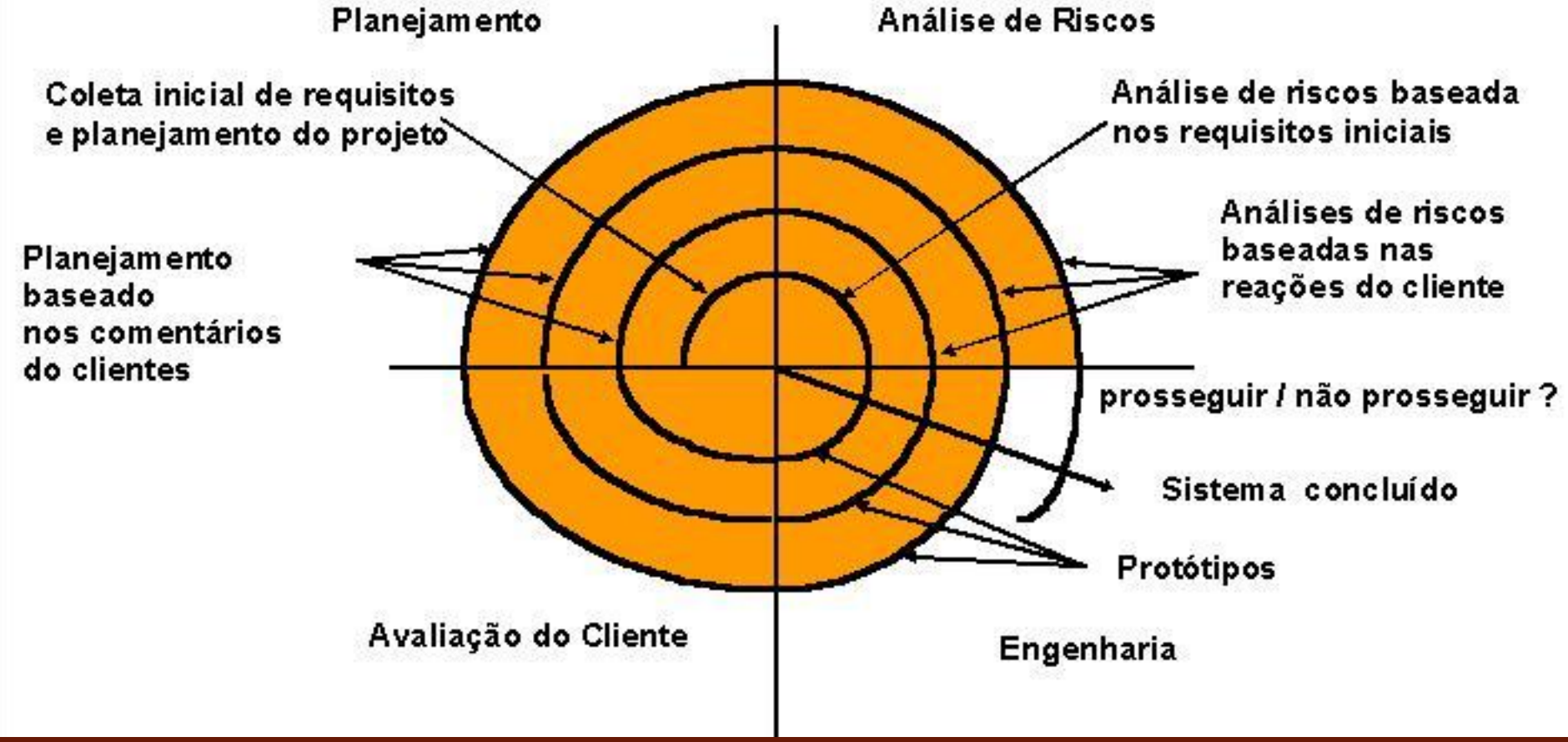
- Para sistemas pequenos
- Requisitos bem compreendidos

Descartável

- Para sistemas maiores
- Requisitos vagos

Prototipagem

- Usuários sentem o sabor do sistema
- Facilita validação de requisitos
- Pode gerar arquiteturas precárias

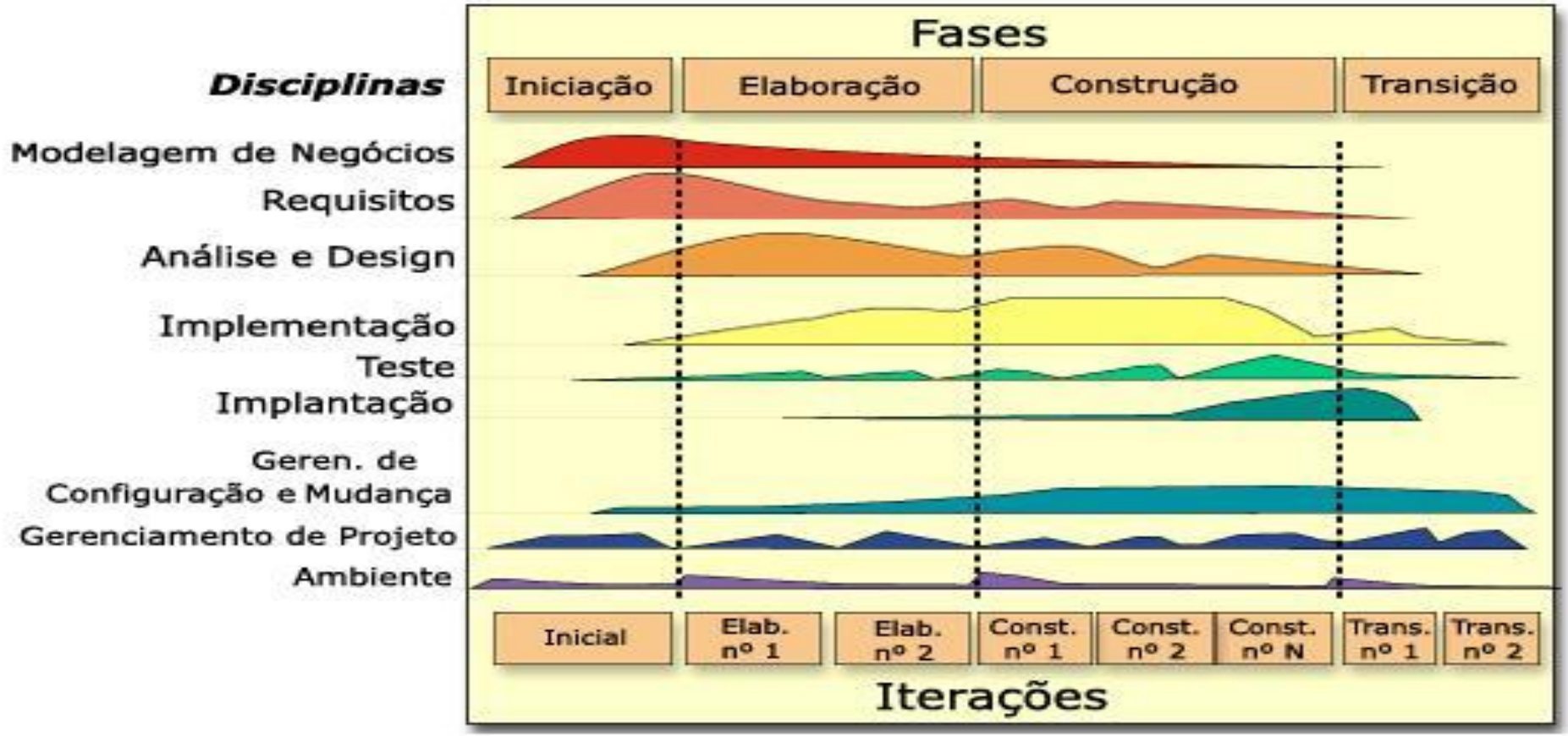


Modelo de Desenvolvimento de SW

Espiral

Palavra chave: RISCO

Modelo espiral



Modelo de Desenvolvimento de SW

RUP

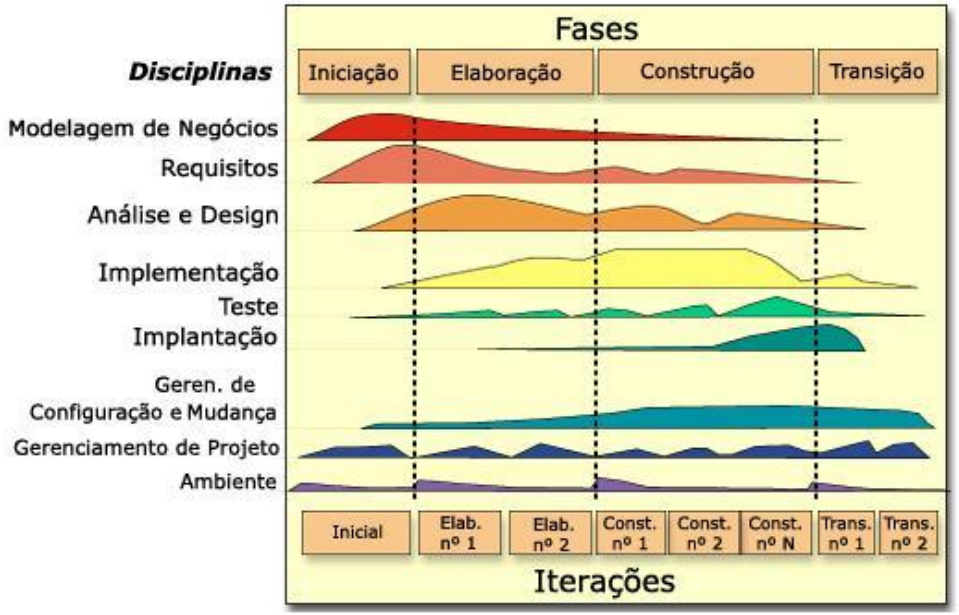
Modelos iterativos - RUP

Fases com atividades sobrepostas.

Forte integração com a UML.

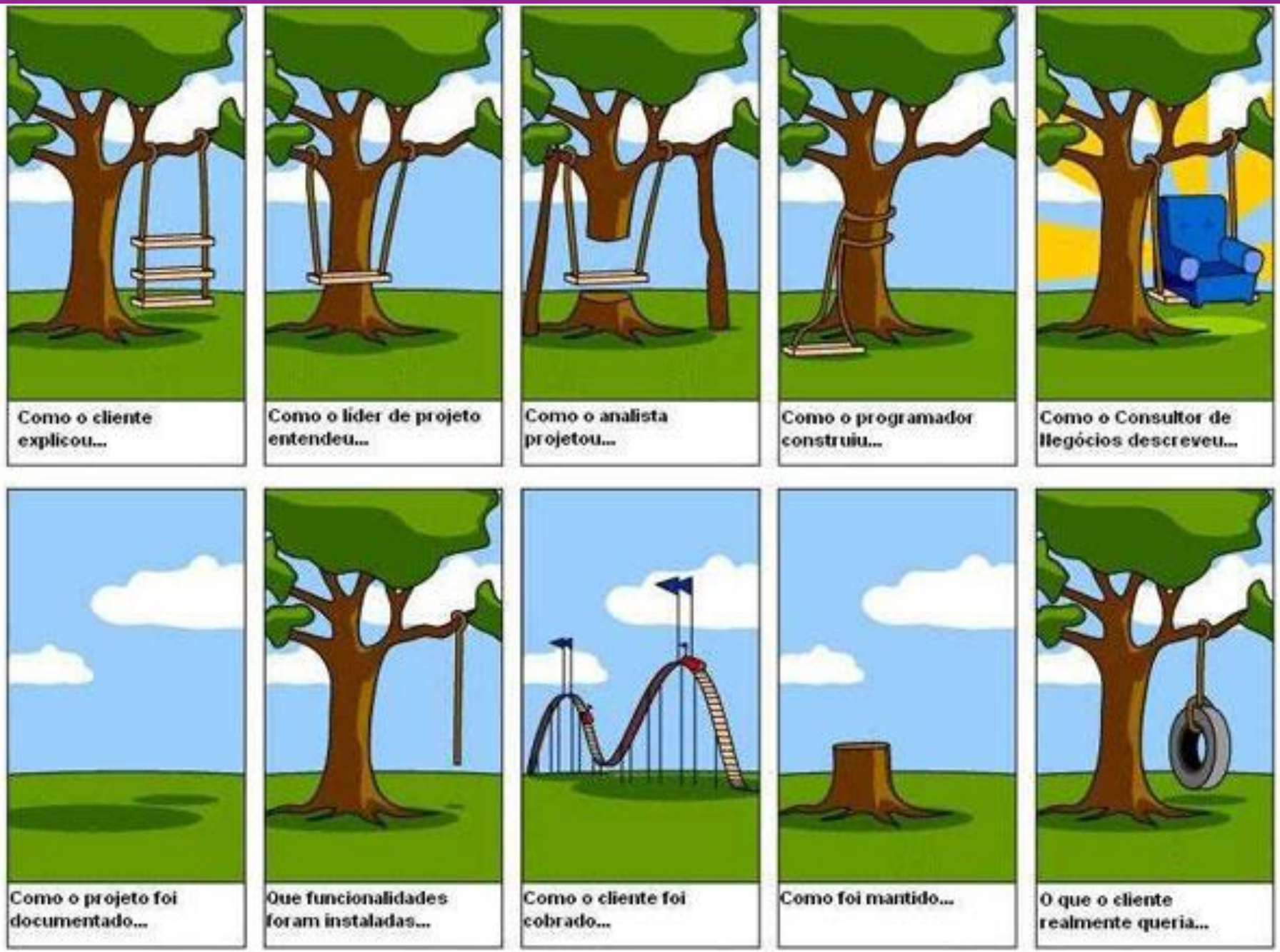
A implantação passa a fazer parte do processo.

Palavra chave: UNIFICADO



RUP – boas práticas

- Desenvolver software iterativamente
- Gerenciar requisitos
- Usar arquitetura baseada em componentes
- Modelar software visualmente
- Verificar a qualidade do software
- Controlar as mudanças do software


Como o cliente explicou...
Como o líder de projeto entendeu...
Como o analista projetou...
Como o programador construiu...
Como o Consultor de Negócios descreveu...
Como o projeto foi documentado...
Que funcionalidades foram instaladas...
Como o cliente foi cobrado...
Como foi mantido...
O que o cliente realmente queria...

Manifesto para o desenvolvimento ágil de software

Estamos descobrindo maneiras melhores de desenvolver software fazendo-o nós mesmos e ajudando outros a fazê-lo. Através deste trabalho, passamos a valorizar:

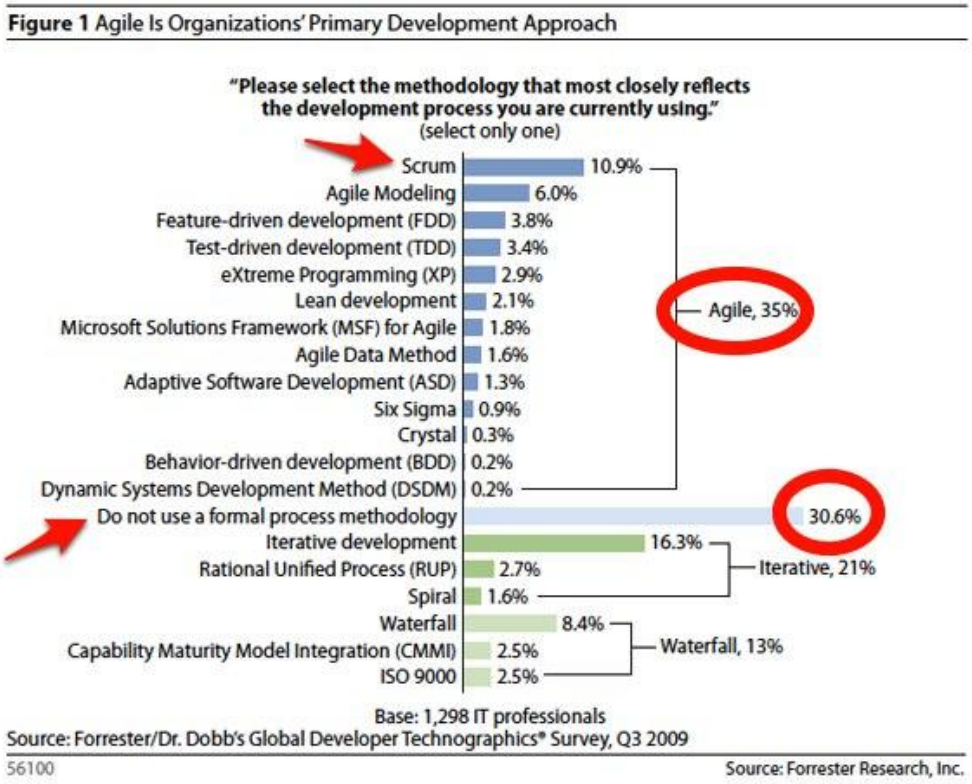
Indivíduos e interação entre eles mais que processos e ferramentas
Software em funcionamento mais que documentação abrangente
Colaboração com o cliente mais que negociação de contratos
Responder a mudanças mais que seguir um plano

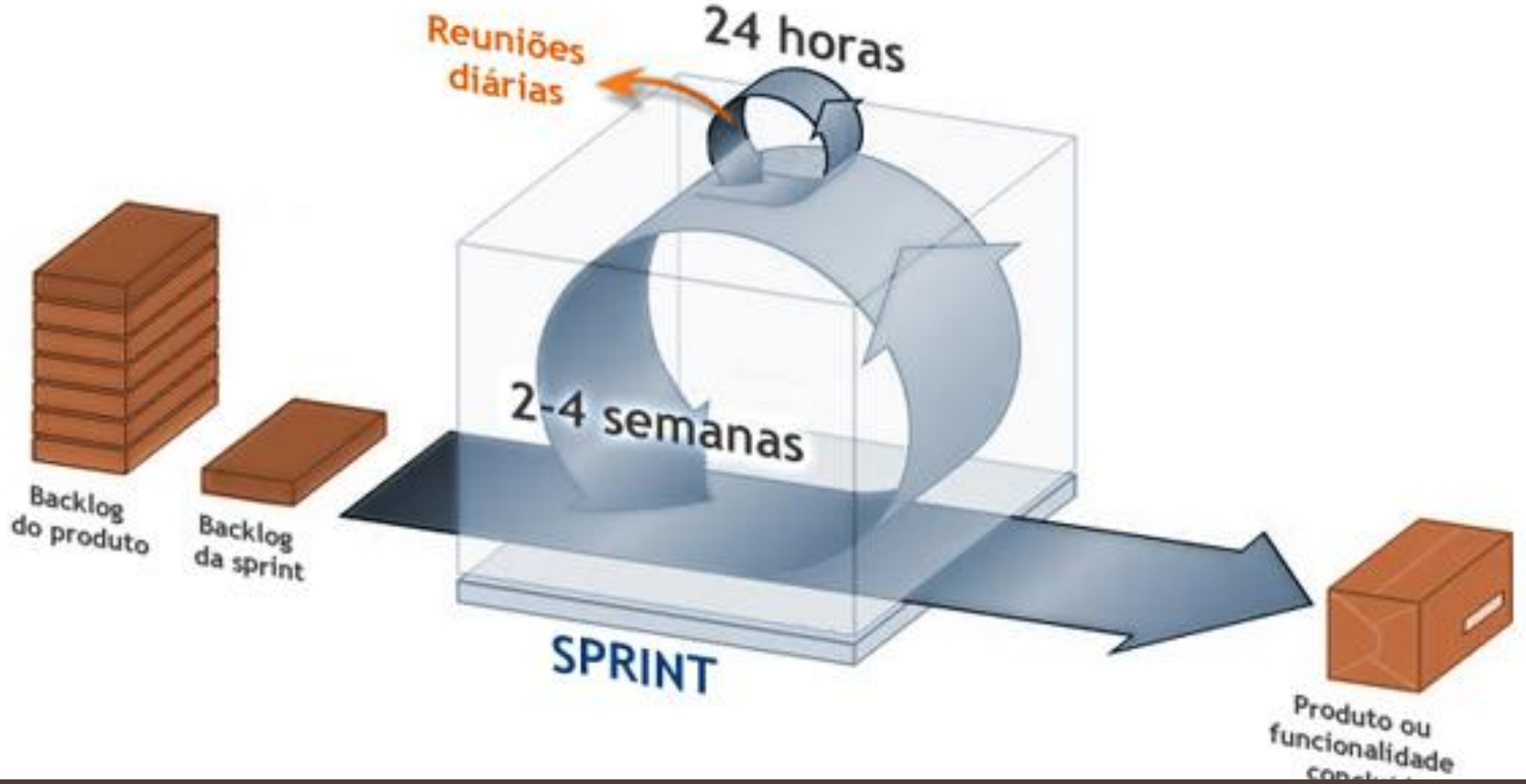
Ou seja, mesmo havendo valor nos itens à direita, valorizamos mais os itens à esquerda.

Kent Beck	Mike Beedle	Arie van Bennekum	Alistair Cockburn
Ward Cunningham	Martin Fowler	James Grenning	Jim Highsmith
Andrew Hunt	Ron Jeffries	Jon Kern	Brian Marick
Robert C. Martin	Steve Mellor	Ken Schwaber	Jeff Sutherland

Metodologias ágeis

http://agilescout.com/learn-more-agile-software-development-methods-this-year/





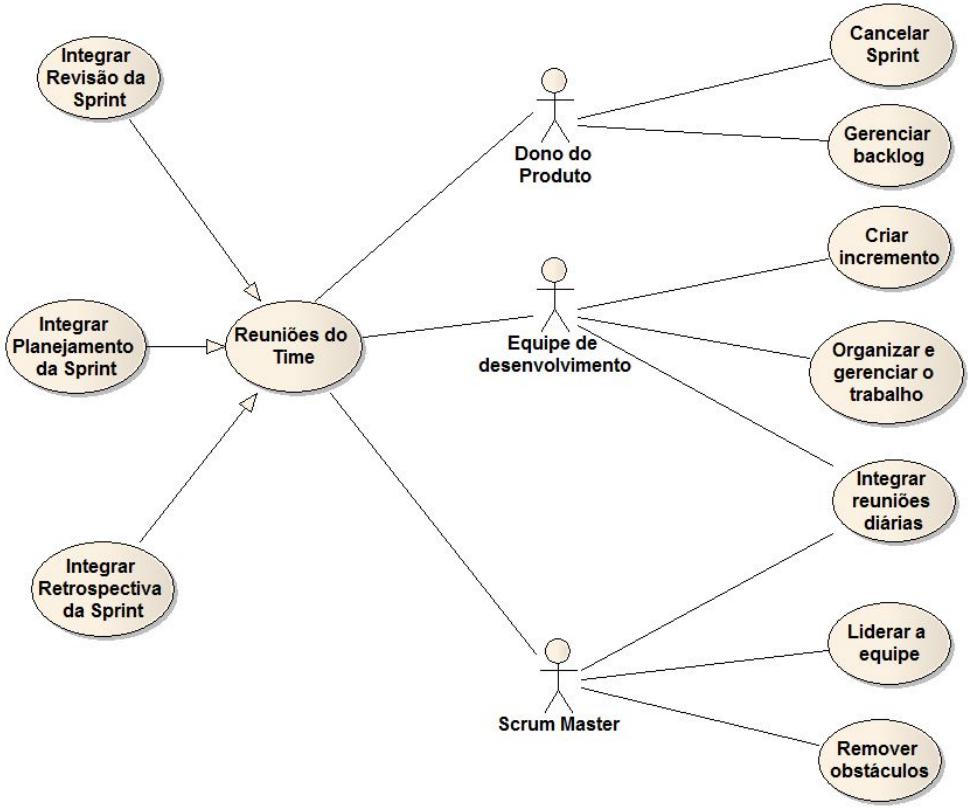
Scrum

http://workinside.com.br/
https://www.scrum.org/

Scrum

- **Framework** de processo ágil
- Para gerenciamento de **equipes**
- Equipes de no máximo 9 pessoas
- Equipes multifuncionais e autogerenciáveis
- Iterações de no máximo 1 mês
- Entrega de incrementos funcionais

Scrum

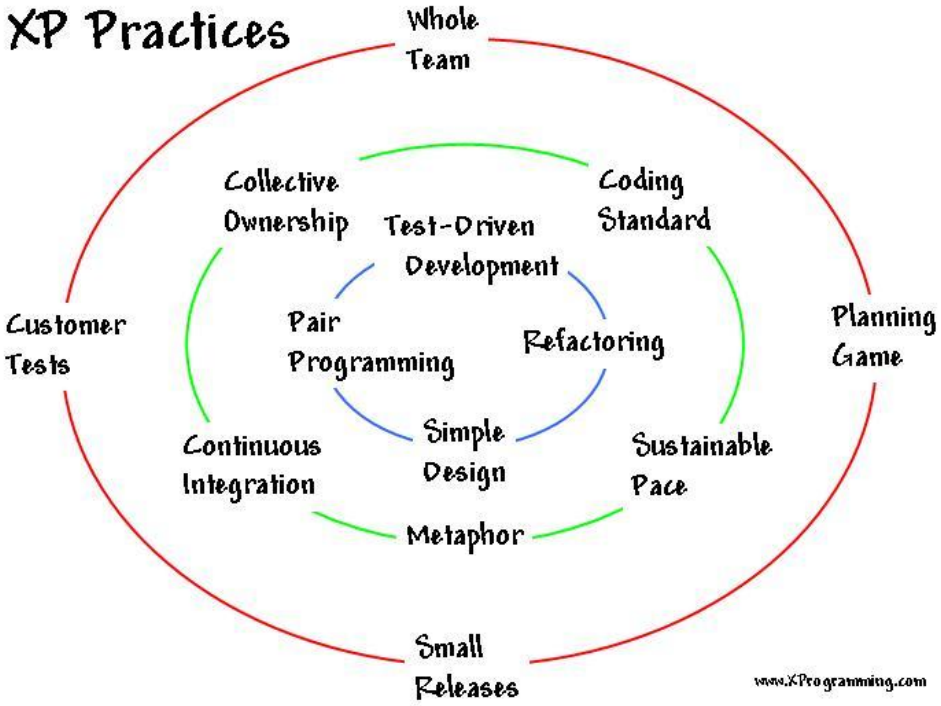

Integrar Revisão da Sprint
Integrar Planejamento da Sprint
Integrar Retrospectiva da Sprint
Reuniões do Time
Dono do Produto
Equipe de desenvolvimento
Scrum Master
Cancelar Sprint
Gerenciar backlog
Criar incremento
Organizar e gerenciar o trabalho
Integrar reuniões diárias
Liderar a equipe
Remover obstáculos

Extreme Programming

XP é uma metodologia leve para times de tamanho pequeno a médio que desenvolvem software em face a requisitos que se modificam rapidamente (Kent Beck).

Extreme Programming

http://xprogramming.com/



Extreme Programming

Por que extremo?

Se revisar o código é bom, revisaremos o código o tempo inteiro.

Se testar é bom, testaremos o tempo inteiro, inclusive os clientes.

Se simplicidade é bom, então faremos tudo o mais simples possível que possa funcionar.

Se arquitetura é bom, refinaremos a arquitetura o tempo inteiro.

Se integração é bom, integraremos o código o tempo inteiro.

Se iterações curtas são boas, faremos iterações muito, mas muito curtas mesmo.

Extreme Programming

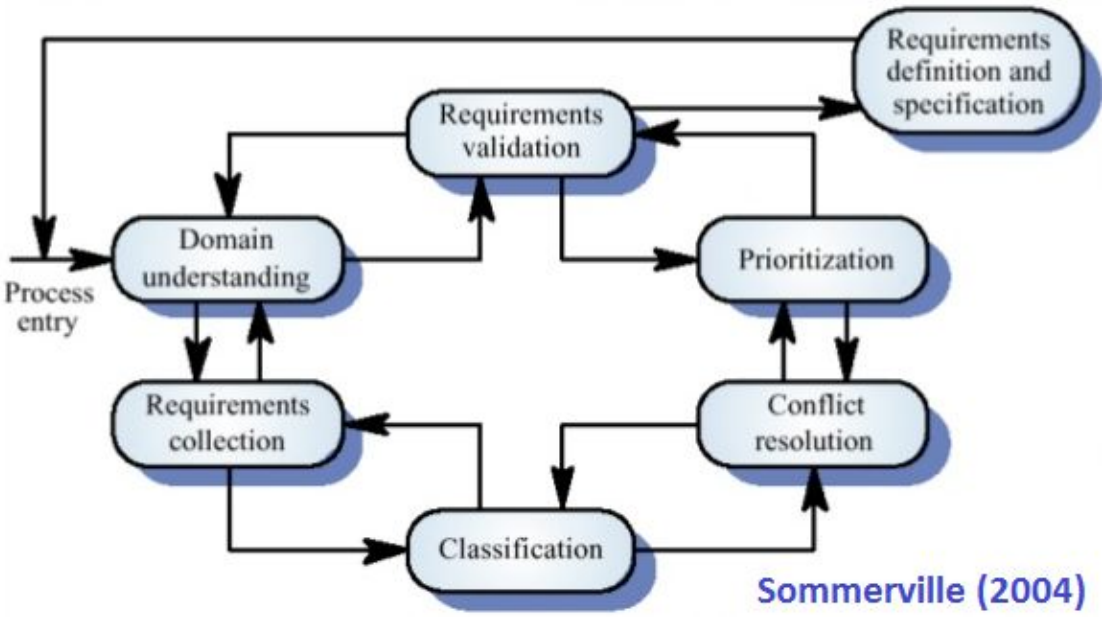
Valores

- Comunicação
- Simplicidade
- Feedback
- Coragem

Atividades básicas

- Codificar
- Testar
- Ouvir
- Projetar

Engenharia de Requisitos



Requisitos

O que é um requisito?

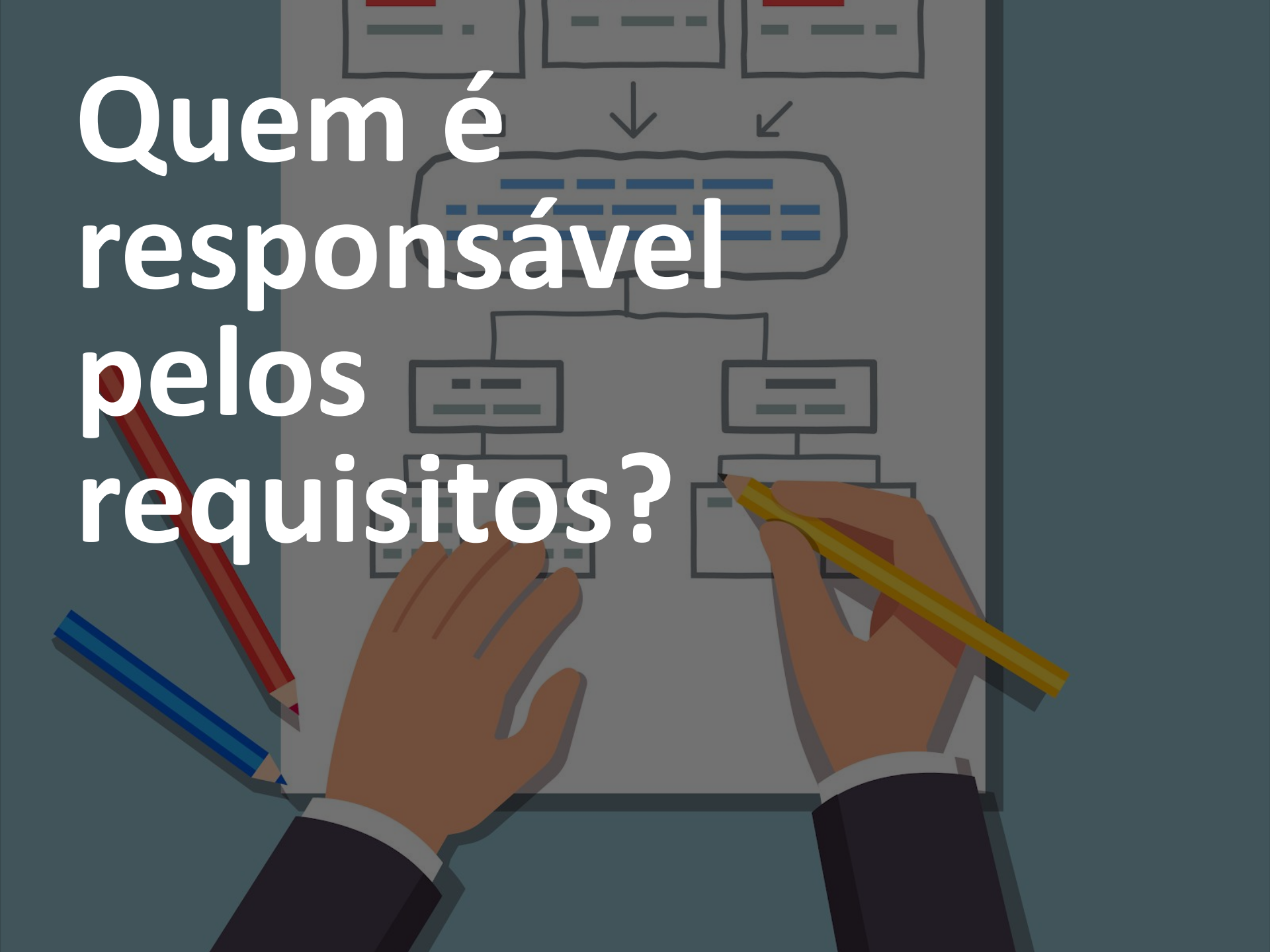
- Característica, atributo, habilidade ou qualidade que um sistema deve prover para **ser útil a seus usuários**

- Os requisitos devem informar "o que deve ser feito ou atendido" **para resolver o problema do usuário**
  - Base para o projeto (design)
  - Mas, não consideramos a solução técnica

Logo...


Scripts SQL
Modelos ER
físicos
Padrões de
projeto
Estruturas
internas de dados

Quem é responsável pelos requisitos?



Quem são os fornecedores de requisitos?

Quem tem conhecimento de uso ou de negócio para orientar a construção do produto?

Quem sabe como o sistema funciona?

Quem conhece as regras do negócio?

Analista de Negócio

Reconhecer os fornecedores de requisitos adequados para o projeto/produto.

Entende as necessidades, expectativas e prioridades do cliente.

Estabelece o *roadmap* do produto.

Funciona como um procurador do cliente para o Analista de Sistemas.

Valida os requisitos com o Analista de Sistemas.

Analista de sistemas

Elicita os requisitos com clientes ou com Analistas de Negócio.

Especifica os requisitos detalhados.

Apresenta os requisitos para a equipe e a apoia nas dúvidas.

Verifica requisitos de outros Analistas de Sistemas (revisão pelos pares).

Valida os requisitos com o Cliente ou Analista de Negócio.

Analista de testes

Participa na reunião de apresentação dos requisitos pelo Analista de Sistemas.

Avalia de modo crítico e aprova os requisitos apresentados pelos Analistas de Sistemas, com foco em viabilizar a preparação dos dados e casos de teste.

Especifica os roteiros de teste a partir dos requisitos detalhados.

Arquiteto

Participa na reunião de apresentação dos requisitos pelo Analista de Sistemas.

Avalia de modo crítico e aprova os requisitos apresentados pelos Analistas de Sistemas, com foco em viabilizar a implementação dos mesmos.

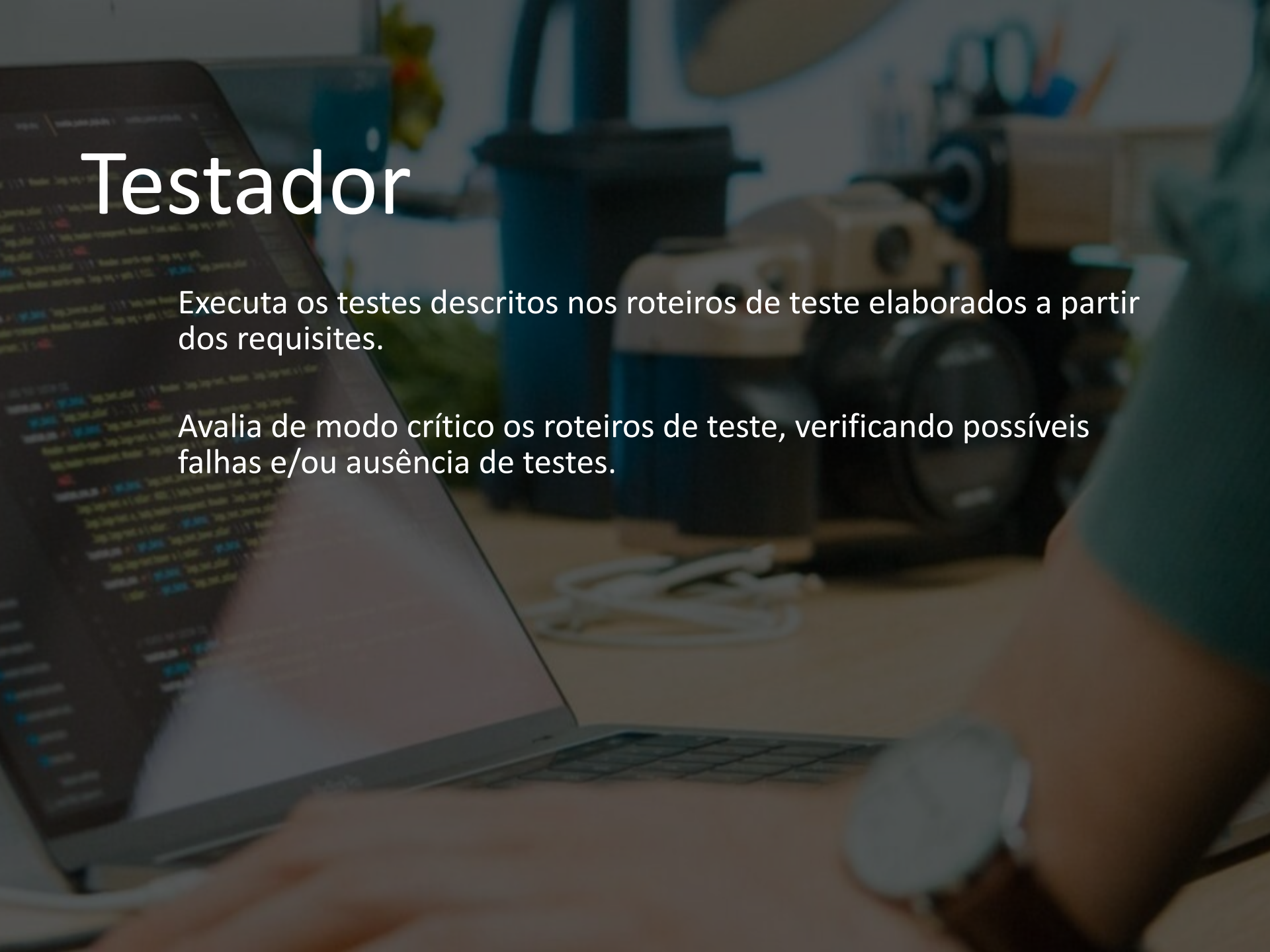
Especifica a arquitetura do produto de acordo com os requisitos aprovados.

Programador

Participa na reunião de apresentação dos requisitos pelo Analista de Sistemas.

Avalia de modo crítico e aprova os requisitos especificados, com foco em identificar a viabilidade de codificação dos mesmos.

Realiza a codificação dos requisitos e os respectivos testes.



Testador

Executa os testes descritos nos roteiros de teste elaborados a partir dos requisites.

Avalia de modo crítico os roteiros de teste, verificando possíveis falhas e/ou ausência de testes.

Como verificar requisitos ?

Critérios!!

Completude
Ambiguidade
Testabilidade
Viabilidade

Critérios de completude

- O requisito está escrito no formato estabelecido?

Verifique também se o requisito representa uma necessidade e não uma solução.

- O requisito está escrito na voz ativa e no afirmativo?

- As palavras-chaves estão associadas ao seu respectivo significado no glossário?

- O requisito é único no sistema?

Aqui, não é apenas uma questão de identificador, mas de conteúdo do requisito.

- O requisito não contradiz outro requisito?

...

Critérios de ambiguidade

O requisito tem uma interpretação única?

- Verifique se é possível interpretar o requisito de mais de uma maneira.

- Verifique se há informações conflitantes no documento.

Para escrever de forma objetiva, verifique palavras suspeitas: Alcançável, adequado, aproximadamente, completo, eficiente, minimizar, maximizar, flexível, modular, nominal, normalmente, etc, otimizado, tipicamente, usualmente, geralmente, frequentemente, fácil, simples, muitos, vários, alguns, poucos, tanto quanto possível, pequeno, grande, baixo, alto, versátil, amigável, escalável, e/ou.

Critérios de testabilidade

O requisito pode ser testado?

- Verifique se o requisito está escrito de modo que seja possível determinar se a funcionalidade estará ou não presente no sistema.

- Verifique se existem cenários ou exemplos que podem ser anexados ao texto do requisito.

Critérios de viabilidade

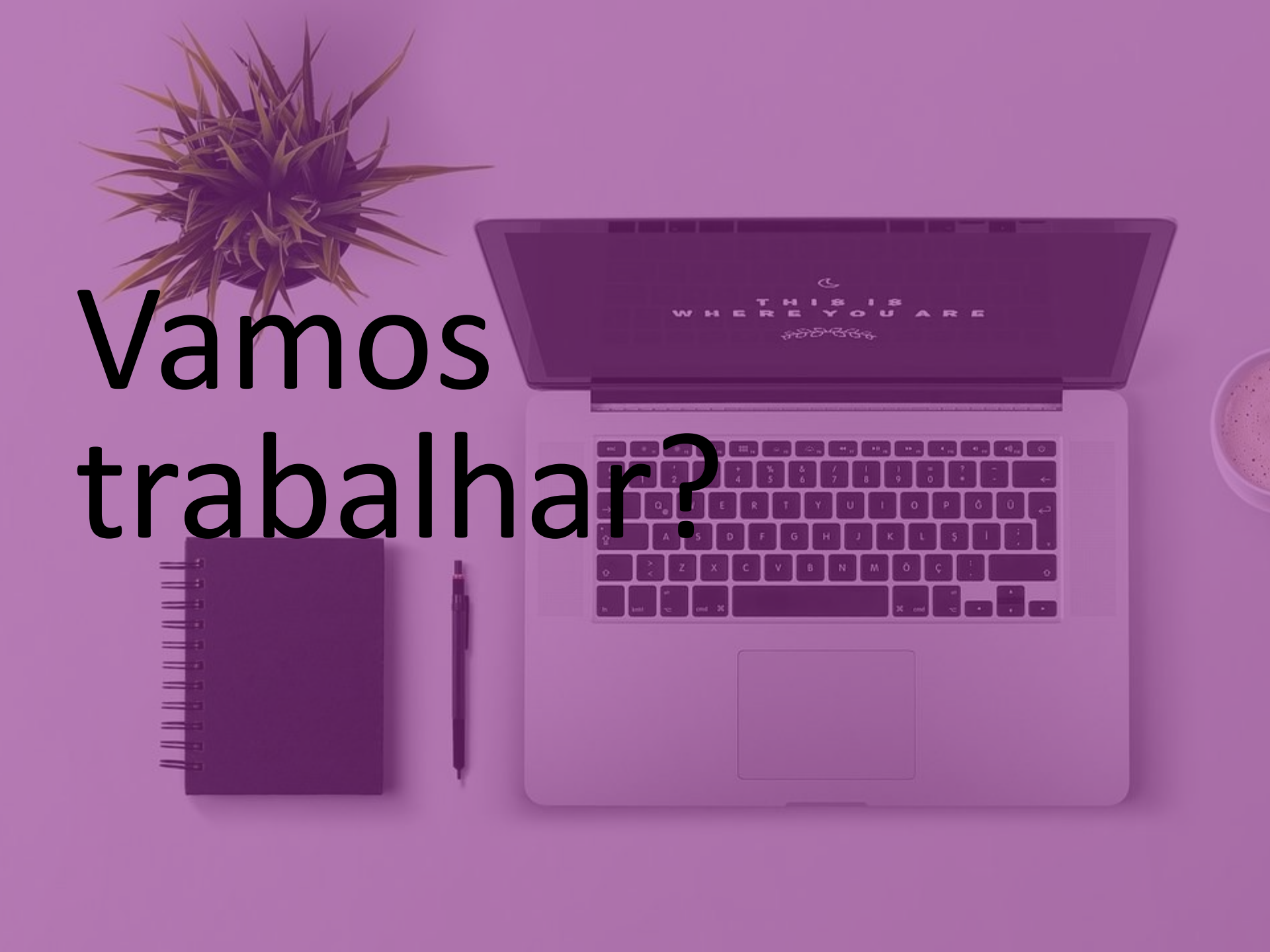
O requisito é viável tecnicamente?

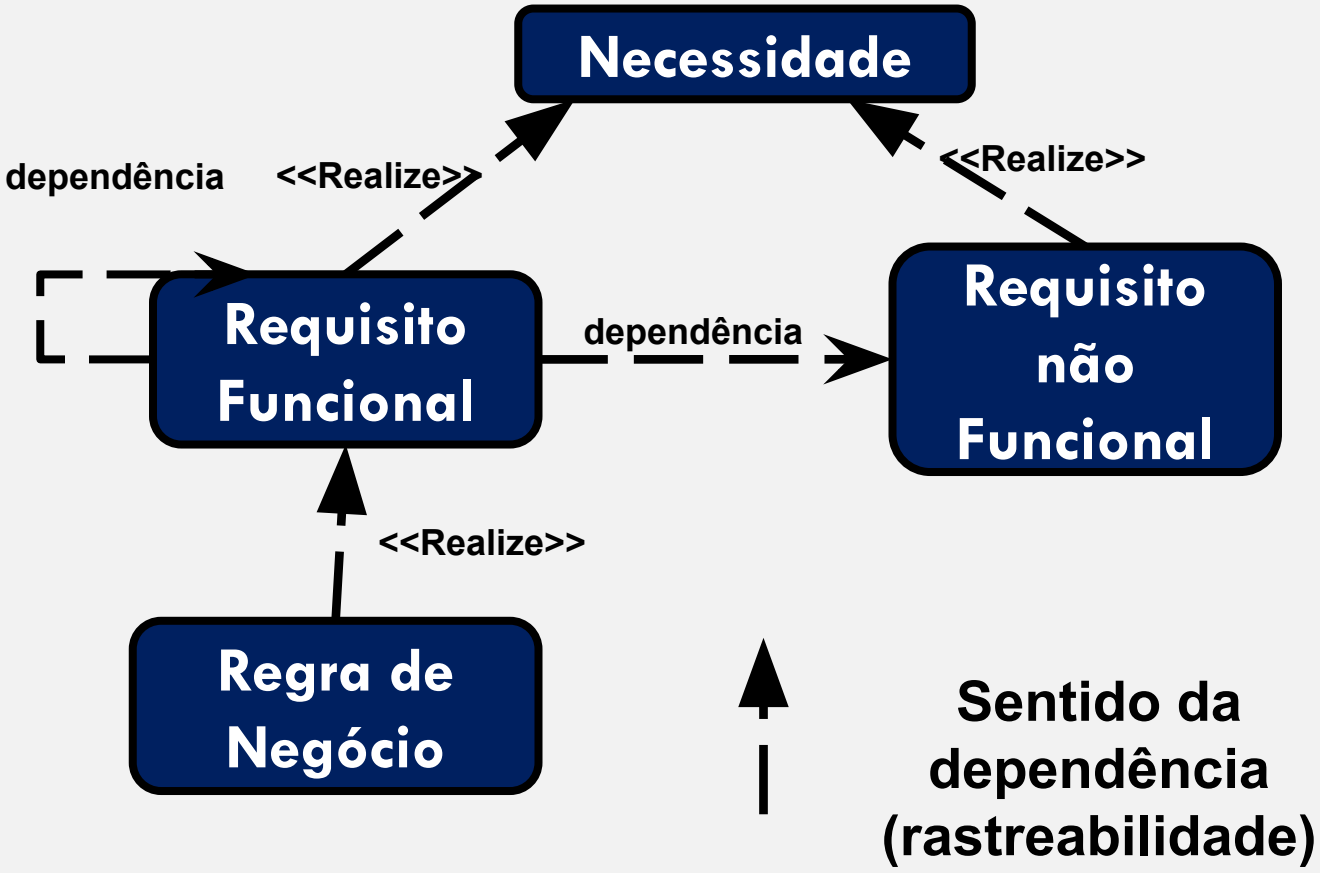
Se você tem dúvidas técnicas, converse com a sua equipe (arquiteto, projetista, programadores).

Verifique padrões de interface gráfica com as orientações e limitações para frameworks utilizados.

Identifique os cenários de infraestrutura para garantir aspectos de desempenho.

Vamos trabalhar?




Necessidade
dependência
<<Realize>>
<<Realize>>
Requisito Funcional
dependência
Requisito não Funcional
<<Realize>>
Regra de Negócio
Sentido da dependência (rastreabilidade)

Necessidade

- Representa algo que o usuário/cliente “precisa” para resolver um problema.
- Declarações com forte significado para o stakeholder.
- Foco naquilo que o sistema irá resolver e não no que o sistema irá fazer.

  Não confundir necessidades com requisitos funcionais

Por que estamos fazendo este trabalho?

Onde e como o resultado do trabalho será **utilizado**?

Quais são os **objetivos de negócio**?

O que o cliente deseja realmente **resolver**?

O que o cliente realmente **precisa**?

O que **incomoda** o cliente?

Você pode usar ***modelagem de negócio*** *para ter uma visão geral do domínio do problema*

Algumas ideias para definir o problema...

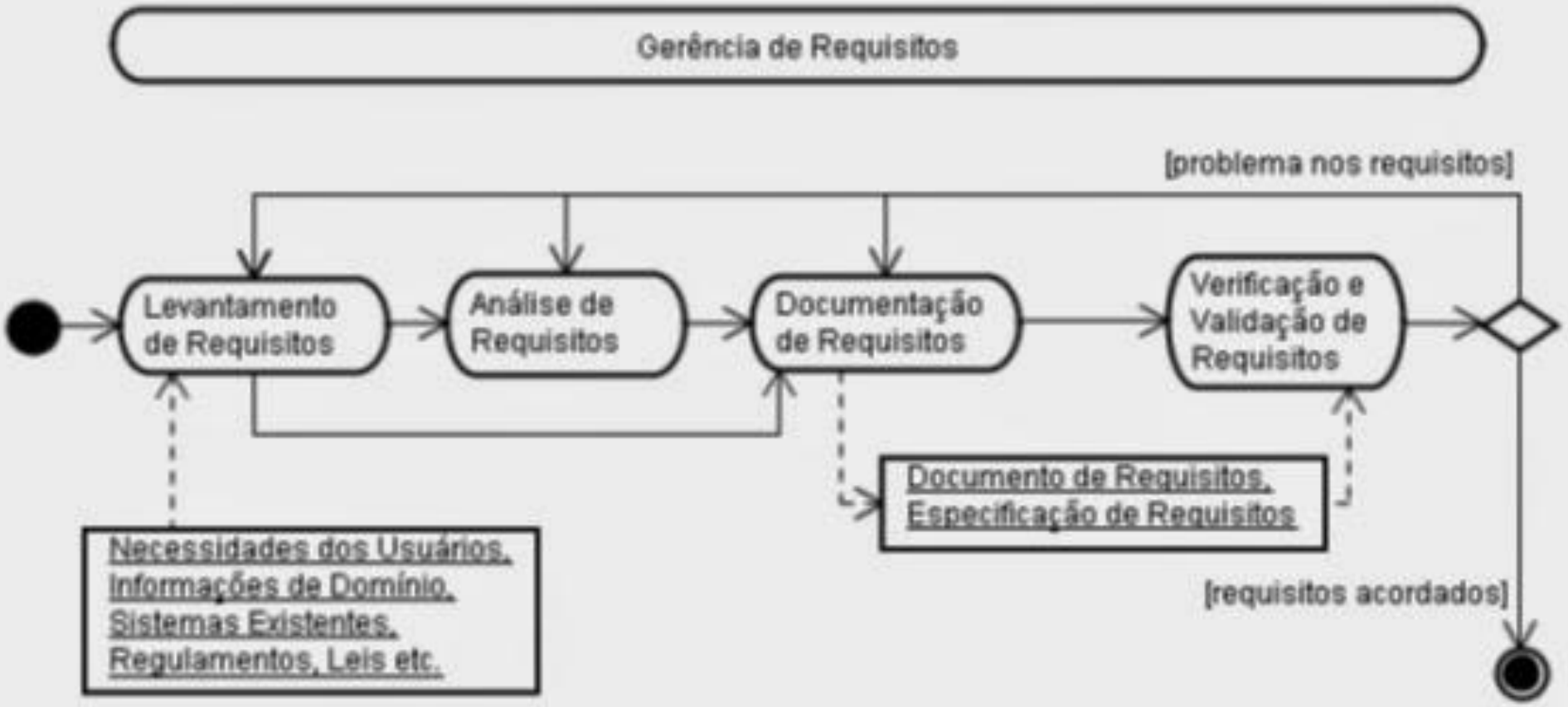
Reduzir o tempo de ...

Aumentar a produtividade ...

Atender mais clientes ...

Aumentar a segurança sobre ...

Diminuir o volume de ...


Gerência de Requisitos
[problema nos requisitos]
Levantamento de Requisitos
Análise de Requisitos
Documentação de Requisitos
Verificação e Validação de Requisitos
Documento de Requisitos, Especificação de Requisitos
Necessidades dos Usuários, Informações de Domínio, Sistemas Existentes, Regulamentos, Leis etc.
[requisitos acordados]

Requisitos

Requisitos

NE<NNN> - <Descrição da necessidade>
onde ***<NNN>*** *representa um número sequencial.*

Vamos analisar este exemplo:

NE001 - O sistema deve processar planilhas financeiras

NE001 - O sistema deve processar planilhas financeiras

- Texto voltado para a solução, se aproximando de um requisito funcional
  - Foco no que deve ser feito e não no problema do usuário

- Por que o usuário/cliente quer processar estas planilhas?
  - A resposta é porque ele precisa calcular as despesas financeiras por categoria da sua empresa

- Qual o benefício que o usuário/cliente terá com o processamento das planilhas?
  - A resposta é que o cálculo será feito automaticamente, aumentando a velocidade de resposta e eliminando erro humano.
  - O usuário também deseja guardar um histórico dos cálculos para conferência futura.

- Existem outros modos de resolver o problema identificado? O processamento das planilhas é uma possível solução, mas é a melhor?

NE001 - Aumentar a velocidade dos cálculos, guardar histórico e eliminar erro humano no cálculo.

Requisitos Funcionais

Requisito Funcional

- Expressam funcionalidades ou serviços que um sistema deve ou pode ser capaz de executar ou fornecer

- Ações:
  - Imprimir, Calcular, Gerar
  - Exportar, Importar, Consultar
  - Mostrar, Enviar, ...

Forma 1	**RF<MM><NNN> - O sistema deve permitir <descrição da função executada por algum usuário do sistema>**
Quando usar	Note que o verbo "**permitir**" deve ser usado aqui para indicar que a função será executada por um usuário do sistema.
Forma 2	**RF<MM><NNN> - O sistema deve <descrição da função executada pelo sistema>**
Quando usar	Note que a ausência do verbo "permitir" indica que a função será executada diretamente pelo sistema, sem a interação direta com um usuário.

Onde:

RF	Prefixo adotado para representar um requisito funcional
<MM>	Número sequencial com dois dígitos, identificando o módulo que o requisito pertence.
<NNN>	Número sequencial com três dígitos, utilizado para a identificação do requisito dentro de cada módulo.

Vamos analisar este exemplo:

RF001 - O sistema deve permitir cadastrar funcionários.

RF01.001 - O sistema deve permitir cadastrar funcionários ativos.

- Qual o significado de termos como “Ativo”, “funcionários”?
- Onde colocar?
- E quando tivermos sinônimos?

RF001 - O sistema deve permitir cadastrar funcionários ativos.

- Lembre-se que estamos representando aqui apenas uma função esperada
- Você ainda terá que trabalhar nas regras de negócio associadas para detalhar o funcionamento desta função do ponto de vista do negócio

Revisando...

RF001 - O sistema deve permitir castrar os funcionários que contados por CLT

Vamos analisar este exemplo:

RF002 - O sistema deve proibir o cadastro de funcionários que não possuam o documento de CPF.

RF002 - O sistema deve proibir o cadastro de funcionários que não possuam o documento de CPF.

- Qual a função aqui?
  - Não há uma função específica, mas sim a verificação de uma condição

- Esta afirmação não se trata de um requisito funcional, mas de uma regra de negócio que condiciona o cadastramento de um servidor

Vamos trabalhar?

Exercício

Tente fazer até aula de segunda-feira….

Elabore os requisitos (Necessidade e Funcionais) para um sistema que tem como objetivo atender a gestão de funcionários de uma empresa.

A empresa registra do funcionário os seguintes dados: o nome do funcionário, o número de horas trabalhadas mensais e o número de dependentes.

A empresa paga R$ 10,00 por hora (valor para cálculo do salário trabalho) e R$ 60,00 por dependente (valor para cálculo do salário família) e são feitos descontos de 8,5% sobre o salário trabalho para o INSS e de 5% sobre o salário trabalho para o imposto de renda.

O sistema apresenta o nome, o salário bruto e o salário líquido do funcionário.

NE

NE001- atender a gestão de funcionários para controle da folha de pagamento.

NE002- Os funcionários desejam organizar seus gastos financeiros, sabendo o salário bruto e salário líquido.

NE003 - Deve funcionar em Android porque todos os funcionários possuem este sistema operacional.

NE004 - Melhor o processo e evitar erro humano no cálculo do salário bruto e líquido

RF

RF001 - O sistema de permitir o cadastro de funcionários, com nome, número de horas trabalhadas mensais e o número de dependentes.

RF002 - O sistema deve realizar o cálculo do salário Bruto.

RF003 - O sistema deve realizar o cálculo do salário líquido.

RF004 - O sistema deve realizar o cálculo do salário trabalho.

RF005 - O sistema deve realizar o cálculo do salário família.

RF006 - O sistema deve mostrar para o funcionário o nome, salário líquido e salário bruto.

RNF

RNF001 - Deve funcionar no Android versão 10 e 11, e dispositivos da Xaomi.

RNF002 - O sistema deve responder tudo em até 5 milisegundos

RNF003 - Os dados terão que ser criptografados

RNF004 - O sistema deve ter uma senha para cada usuário par acesso

RN001 - O cálculo do salário trabalho deve ser da seguinte forma:

salárioTrabalho= (horastrabalhadas*10,00)

Exemplo, considerando horastrabalhadas=10. O resultado do salário trabalho será de 100

RN002 - O cálculo do salário família deve ser da seguinte forma:

salarioFamilia= numerodependentes*60

Exemplo, considerando o numerodependentes=2, o calculo seria: salarioFamilia= 2*60 - O resultado do salário Família será de 120

RN003 - O cálculo do salário bruto deve ser da seguinte forma:

salárioobruto= salarioTrabalho+SalarioFamilia

Exemplo, considerando salarioTrabalho=100 e SalarioFamilia=120, o calculo seria: salariobruto= 100+120 - O resultado do salário bruto será de 220

RN004 - O cálculo do salário líquido deve ser da seguinte forma:

salarioliquido= salariobruto- ((salarioTrabalho*0,085) + (salarioTrabalho*0,05))

Exemplo, considerando o salario bruto de 220 reais e o salario trabalho=100, o calculo seria: salarioliquido= 220 - ((100*0,085) + (100*0,05)) - O resultado do salário líquido será de 206,5

Leitura da Semana
Busca ATIVA

Ulife Bibliotecas Online Biblioteca Person

Procure por Engenharia de Software

Abra o livro de Ian Sommerville,

Engenharia de Software 10ª edição

E leia da página 3 até a página 57