QUALIDADE DE SOFTWARE

João Choma Neto

joaochoma+aulas@gmail.com

https://github.com/JoaoChoma

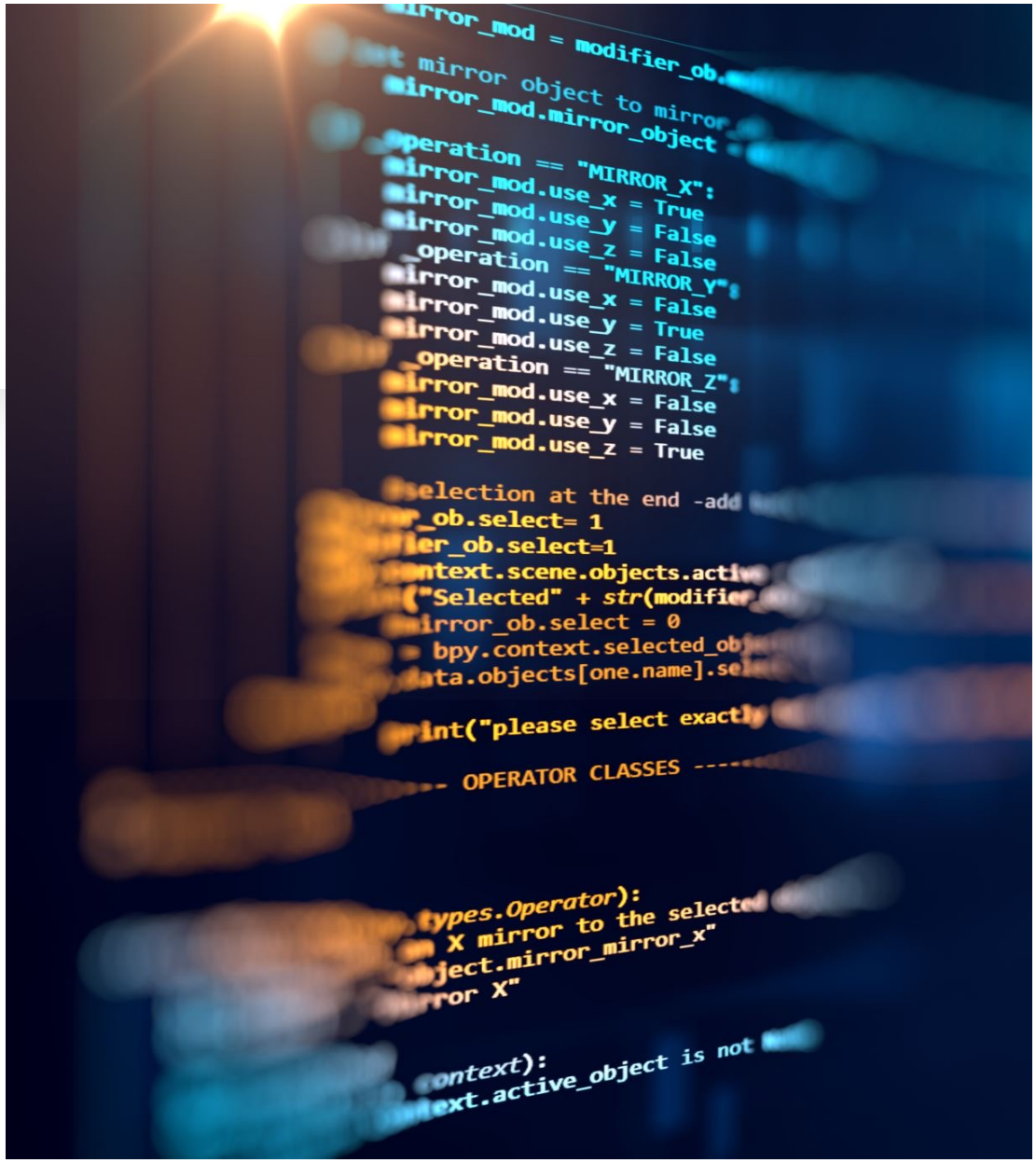
https://github.com/JoaoChoma/qualidadesoftware

ROTEIRO

- Qualidade
- Produto
- Processo

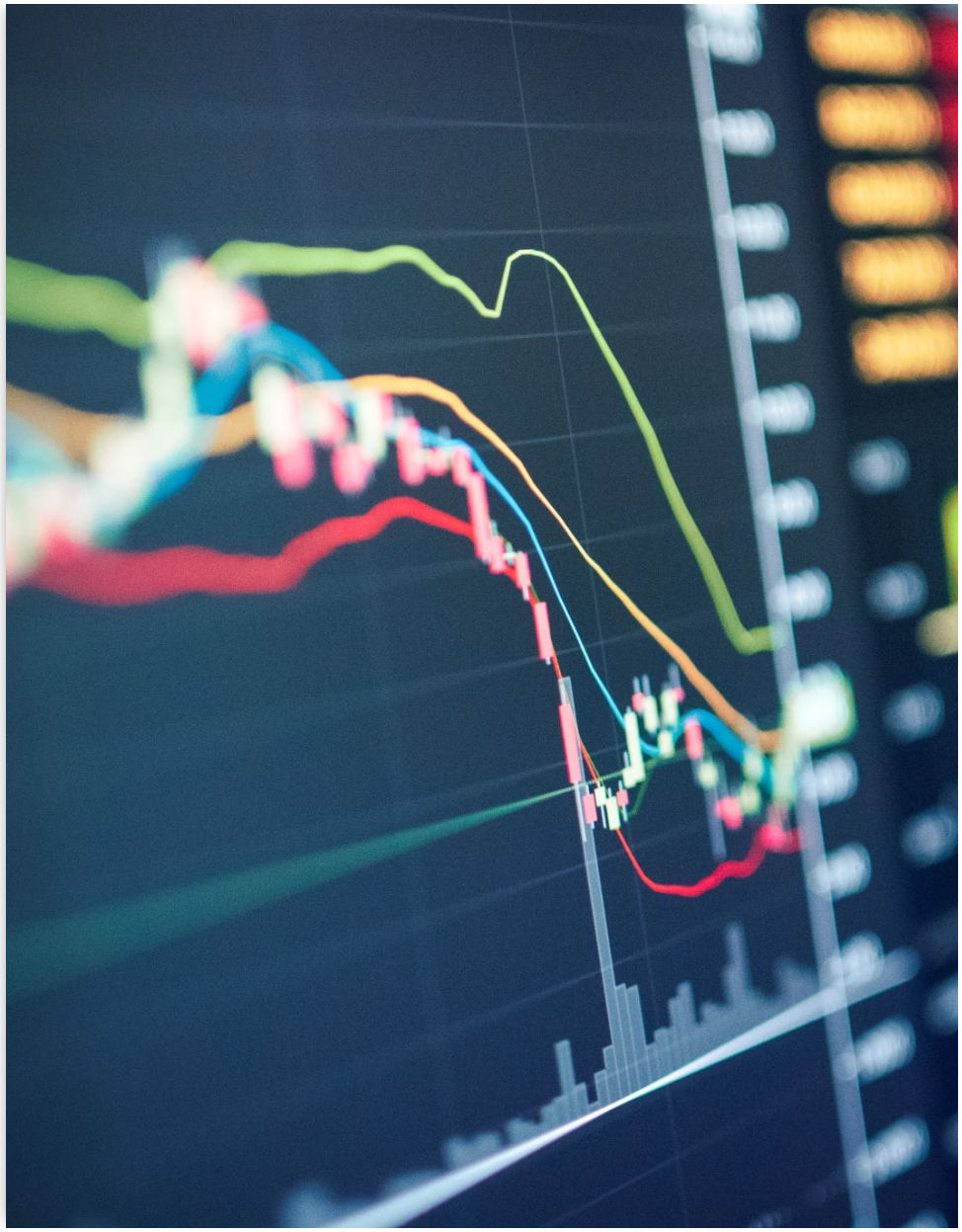
SOFTWARE

- É essencialmente um conjunto de instruções codificadas que comanda o hardware do computador a executar operações



PRODUTO DE SOFTWARE

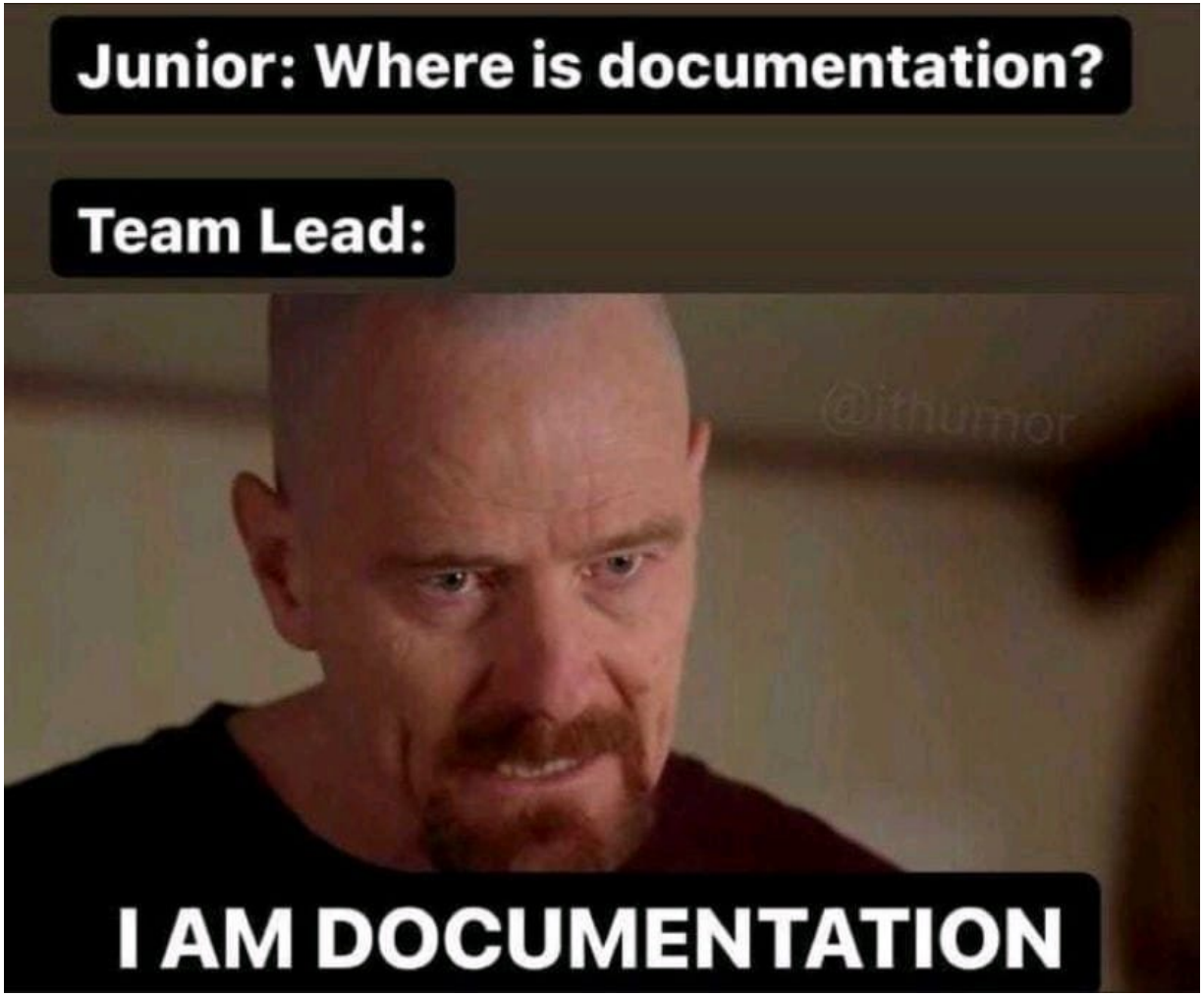
- Um produto de software é geralmente empacotado, comercializado e suportado por uma empresa, visando atender a uma necessidade de mercado específica.



LÁ ANTIGAMENTE

- Não basta vender barato, as novas regras de mercado são orientadas à produção de bens e serviços com qualidade, prazo de entrega determinado, atendimento correto, além de um baixo custo” (Werneck 1994)

- Em algum momento aprendemos sobre a necessidade de documentar um produto de software




100% QUALITY

O QUE É QUALIDADE?

- Qualidade é um conceito amplo que pode variar dependendo do contexto, mas em geral, refere-se à medida em que um produto, serviço, ou processo atende ou excede as expectativas e necessidades dos consumidores ou usuários.

QUALIDADE

- A ISO (International Organization for Standardization) define um conjunto de atributos da qualidade que são amplamente reconhecidos na indústria de desenvolvimento de software
- Esses atributos são essenciais para garantir que o software atenda às expectativas dos usuários e seja considerado de alta qualidade

ATRIBUTOS DE QUALIDADE

- **Funcionalidade:**
  - Refere-se à capacidade do software de fornecer as funções necessárias para atender aos requisitos especificados.
- **Confiabilidade:**
  - Diz respeito à capacidade do software de desempenhar suas funções conforme esperado, mesmo em condições adversas.
- **Usabilidade:**
  - Refere-se à facilidade de uso do software. Um software usável deve ser intuitivo, amigável e eficiente, permitindo que os usuários realizem suas tarefas de forma rápida e sem dificuldades desnecessárias.

ATRIBUTOS DE QUALIDADE

- **Eficiência:**
  - Diz respeito ao desempenho do software em relação aos recursos utilizados. Um software eficiente deve realizar suas funções de maneira rápida e com um consumo adequado de recursos, como CPU, memória e largura de banda.
- **Manutenibilidade:**
  - Refere-se à facilidade com que o software pode ser modificado, corrigido e aprimorado ao longo do tempo. Um software mantível deve ter um código limpo, bem documentado e seguir boas práticas de desenvolvimento.

ATRIBUTOS DE QUALIDADE

- **Portabilidade:**
  - Diz respeito à capacidade do software de ser executado em diferentes ambientes, como diferentes sistemas operacionais ou plataformas de hardware.
- **Adaptabilidade:**
  - Refere-se à capacidade do software de se adaptar a mudanças nos requisitos e no ambiente.

QUALIDADE

- Algo tem qualidade com base em critérios que dizem:
  - tem qualidade
  - não tem qualidade.

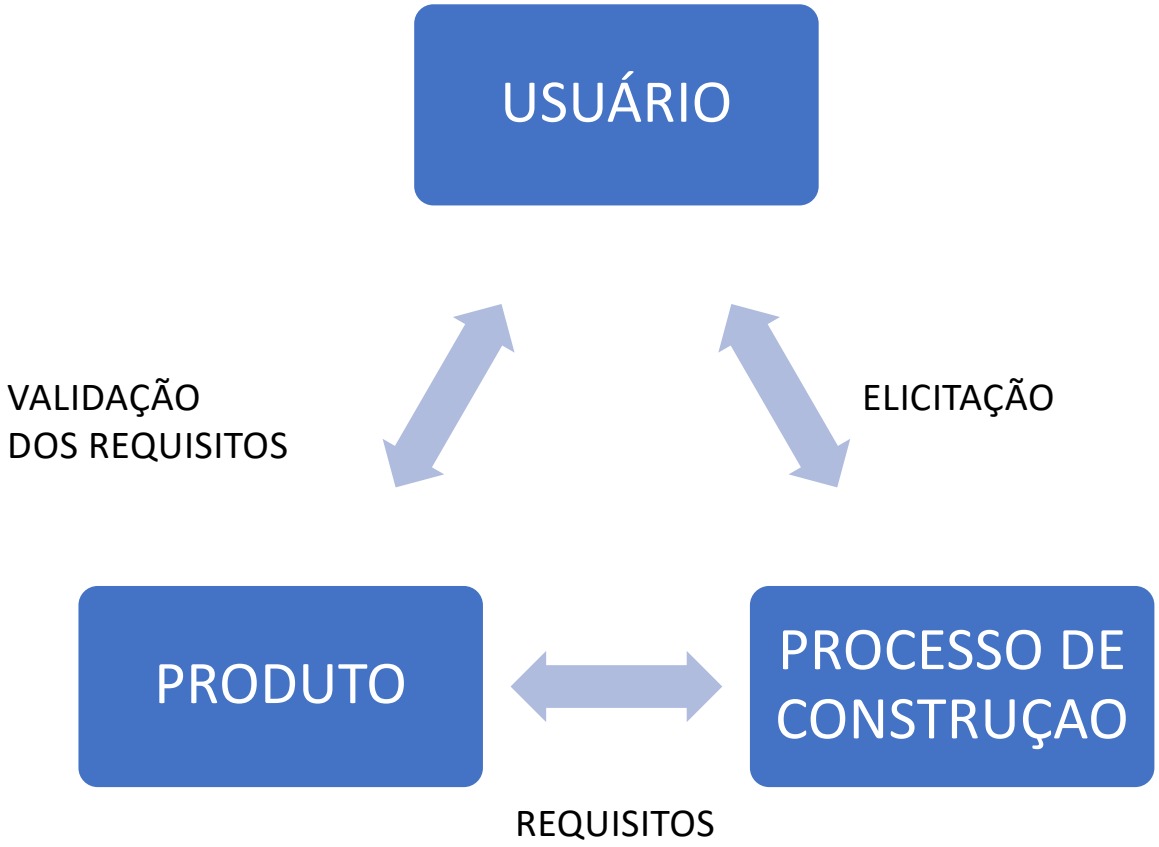
Visão Popular

- Algo abstrato
- Perfeição
- Luxo e questão de gosto

Visão Profissional

- Conformidade aos requisitos
- Adequação ao uso
- Aprovação aos critérios

VISÃO PROFISSIONAL



SEGUNDO A LITERATURA

- Um produto de software apresenta qualidade dependendo do grau de satisfação das necessidades dos clientes sob todos os aspectos do produto” [Sanders, 1994]

SEGUNDO A LITERATURA

- “Qualidade de software é a conformidade a requisitos funcionais e de desempenho que foram explicitamente declarados, a padrões de desenvolvimento claramente documentados, e a características implícitas que são esperadas de todo software desenvolvido por profissionais” [Pressman].

ASPECTOS IMPORTANTES

- Os requisitos de software são a base a partir da qual a qualidade é medida. A falta de conformidade aos requisitos significa falta de qualidade.

ASPECTOS IMPORTANTES

- Padrões especificados definem um conjunto de critérios de desenvolvimento que orientam a maneira segundo a qual o software passa pelo trabalho de engenharia. Se os critérios não forem seguidos, o resultado quase que seguramente será a falta de qualidade.

ASPECTOS IMPORTANTES

- Existe um conjunto de requisitos implícitos que frequentemente não são mencionados na especificação (por exemplo o desejo de uma boa manutenibilidade).

PONTOS DE VISTA

PONTO DE VISTA DO USUÁRIO

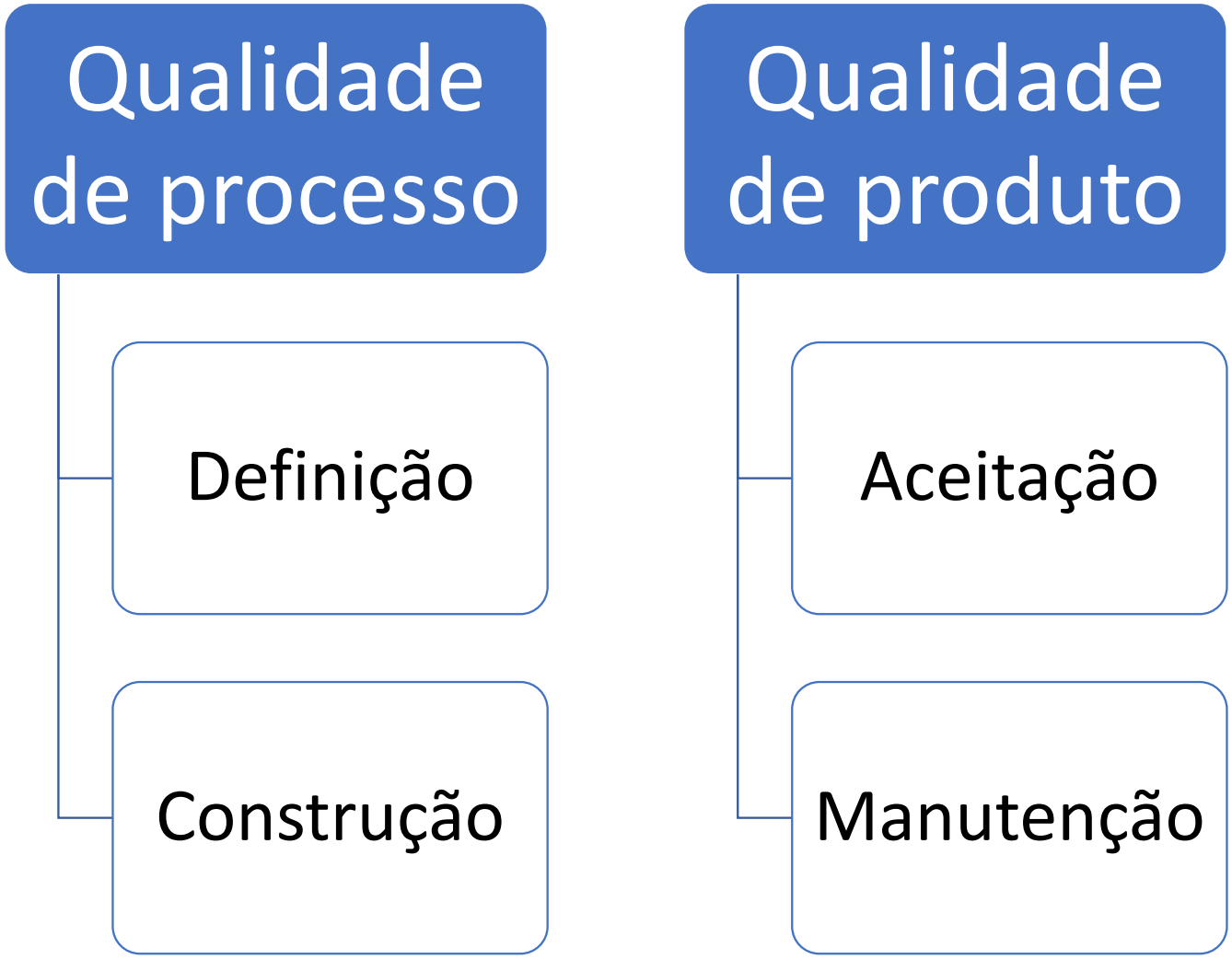
- O interesse fica concentrado principalmente no uso do software
- Avalia o software sem conhecer seus aspectos internos, está apenas interessado na facilidade do uso
- Considera, também, desempenho, na confiabilidade dos resultados e no preço.

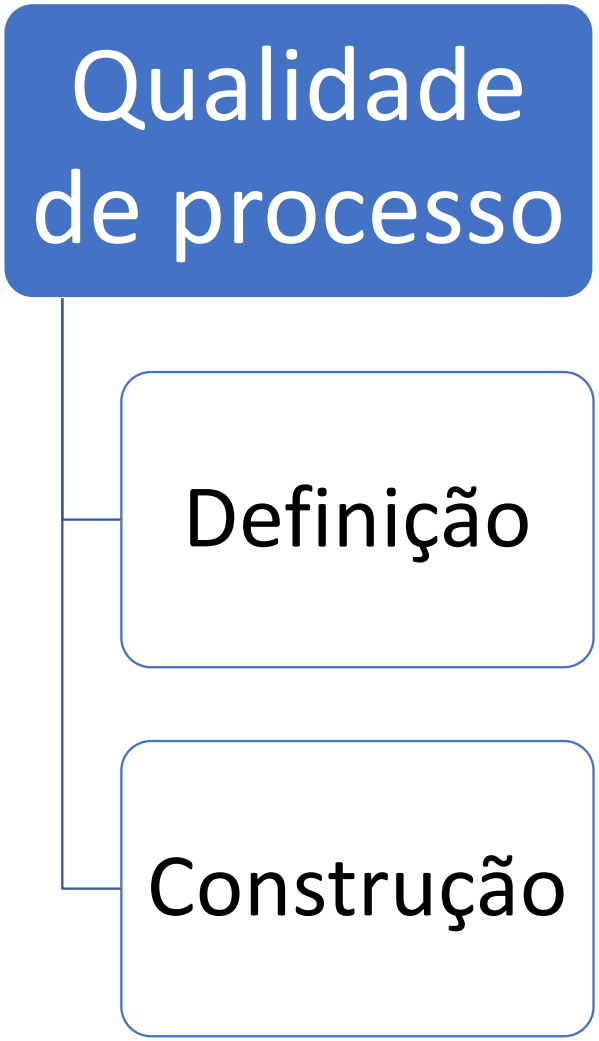
PONTO DE VISTA DO DESENVOLVEDOR

- A qualidade fica voltada as características internas do software
- Avalia aspectos de conformidade em relação aos requisitos do produto
- Considera, também, os aspectos internos do software

PONTO DE VISTA DA ORGANIZAÇÃO

- A qualidade está vinculada aos interesses da organização
- Avalia aspectos de conformidade em relação aos requisitos do produto
- Considera, também, aspectos internos do software
- Considera, também, impacto no processo da organização
- Considera, também, impacto nos resultados da organização


Qualidade de processo
Definição
Construção
Qualidade de produto
Aceitação
Manutenção

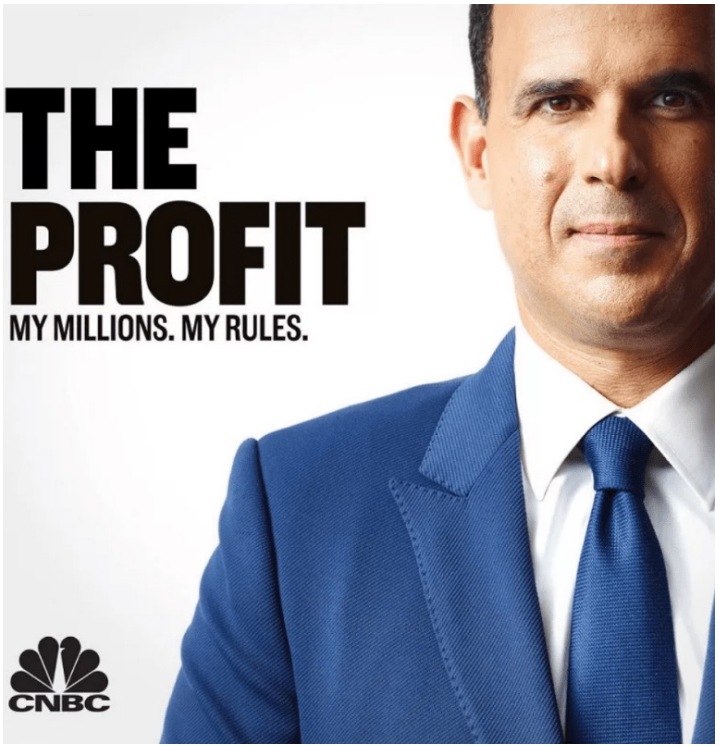

Qualidade de processo
Definição
Construção

QUALIDADE DO PROCESSO

- Esperamos que o aprimorando do processo resulte em um produto de melhor qualidade
- Isso se baseia no princípio de que "qualidade é planejada, projetada e construída", e não apenas inspecionada no produto final

O QUE É UM PROCESSO?

- Um "processo" é uma sequência estruturada de atividades realizadas para alcançar um objetivo específico
- Processos envolvem a transformação de insumos (entradas) em produtos (saídas), usando recursos eficientemente para criar valor



UM PROCESSO TEM

- Um processo possui atividades bem definidas, com começo, meio e fim claros. Cada etapa é mapeada e documentada, permitindo compreensão e repetição.
- Para gerenciar e melhorar um processo, é essencial que ele possa ser medido. Isso inclui a avaliação do desempenho através de métricas específicas, como tempo, custo, qualidade e satisfação do cliente.

UM PROCESSO TEM

- Cada processo é criado para atingir objetivos específicos. Esses objetivos devem ser claros para garantir que o processo seja eficaz e alinhado com as metas mais amplas da organização ou projeto.

UM PROCESSO TEM

Processos são frequentemente projetados para serem repetidos, permitindo a produção de resultados consistentes e previsíveis

Um processo deve ser passível de análise e melhoria. A otimização pode envolver a redução de desperdícios, melhorando a eficiência, ou aprimorando a qualidade das saídas

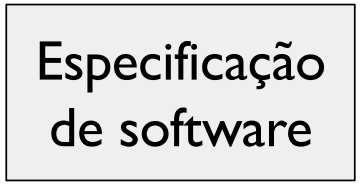
EM SOFTWARE

- **Processos de Desenvolvimento de Software**:
  - Planejamento
  - Desenvolvimento
  - Teste
  - Manutenção

. **Como desenvolver sistemas?**

Processo de Software

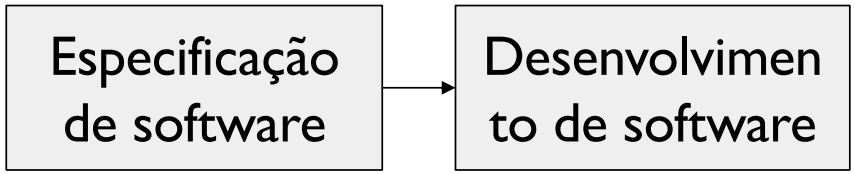
- **Processo de software:** conjunto de atividades e resultados associados que produz um produto de software.



- **Especificação de software:** o software a ser produzido e as restrições para a sua operação são definidos.

Processo de Software

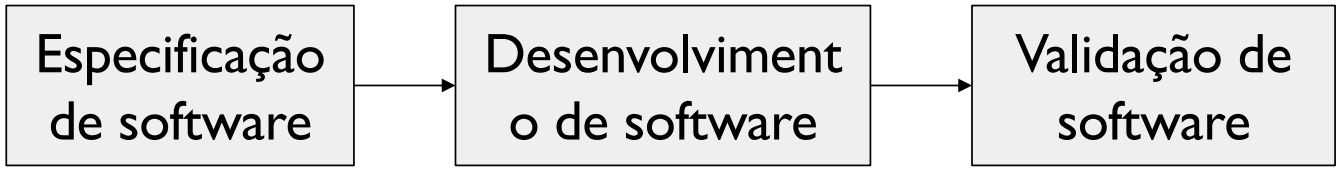
- **Processo de software:** conjunto de atividades e resultados associados que produz um produto de software.



- **Desenvolvimento de software:** o software é projetado e programado.

Processo de Software

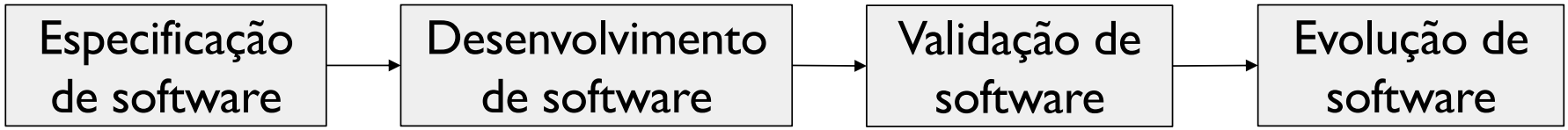
- **Processo de software:** conjunto de atividades e resultados associados que produz um produto de software.



- **Validação de software:** o software é verificado para garantir que é o que o cliente deseja.

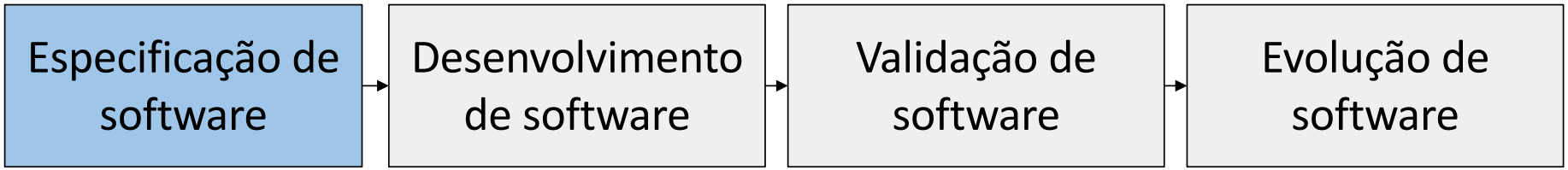
Processo de Software

- **Processo de software:** conjunto de atividades e resultados associados que produz um produto de software.



- **Evolução do software:** o software é modificado de acordo com os novos requisitos do cliente e/ou do mercado.

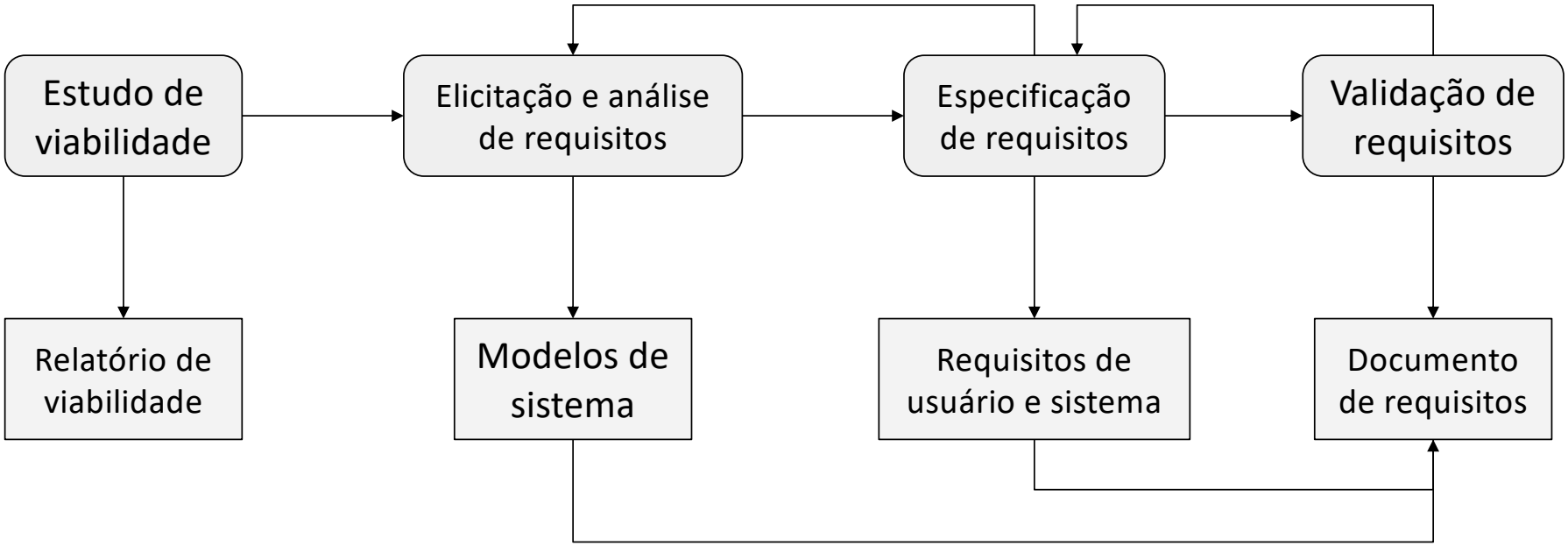
Processo de Software



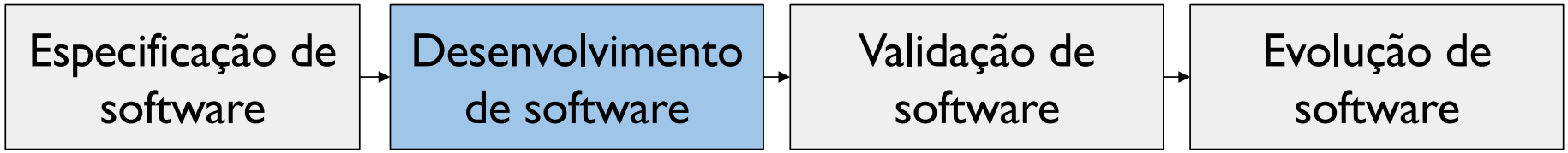
Especificação de Software

- Também conhecida como Engenharia de Requisitos.
- É o processo para compreender e definir quais são as **funcionalidades** necessárias e identificar as **restrições** de operação.
- Etapa crítica do processo de software, pois erros nesse estágio conduzem inevitavelmente a problemas no projeto e na implementação.
- O resultado é um **documento de requisitos**, que é a **especificação do sistema**.

Especificação de Software

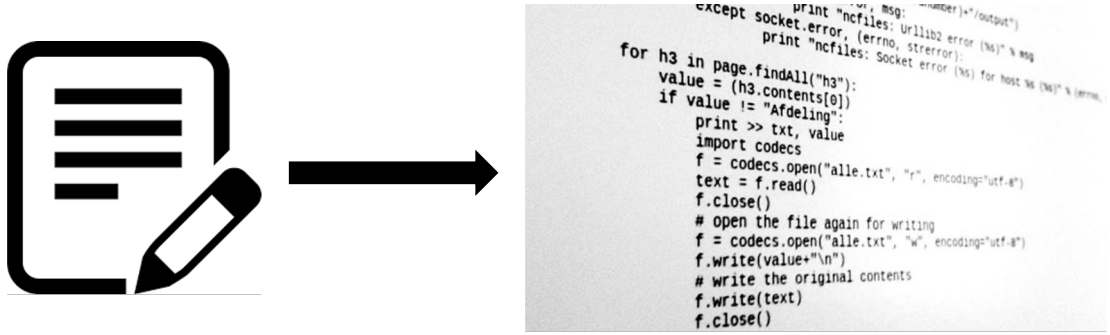


Processo de Software



Projeto e implementação

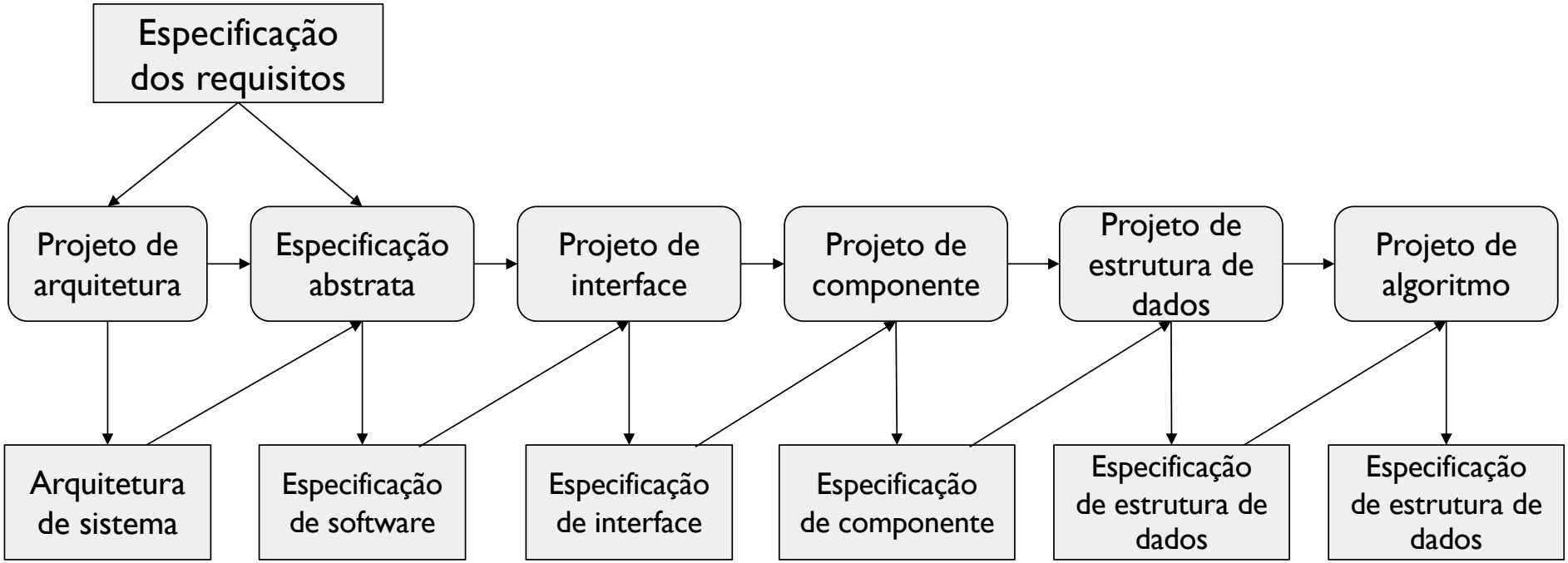
- A etapa de desenvolvimento de software corresponde ao processo de conversão de uma especificação em um sistema executável.
- Envolve os processos de projeto e programação de software, além do refinamento da especificação de software (modelo evolucionário).



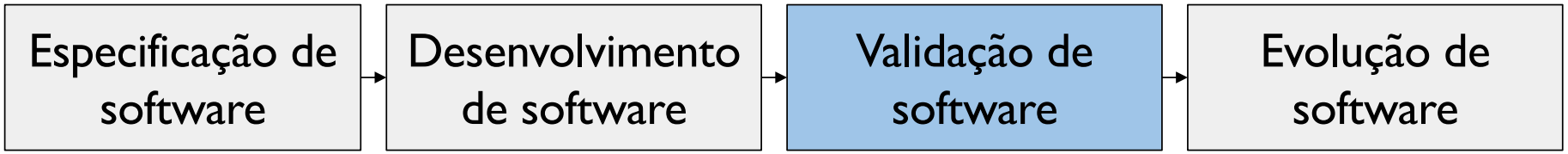
Projeto e implementação

- **Projeto de software** é a descrição da estrutura de software a ser implementada.
  - Dados do sistema,
  - Interfaces entre os componentes,
  - Algoritmos,
  - Outros.
- Desenvolvimento de vários modelos do sistema em diferentes níveis de abstração.

Projeto e Implementação



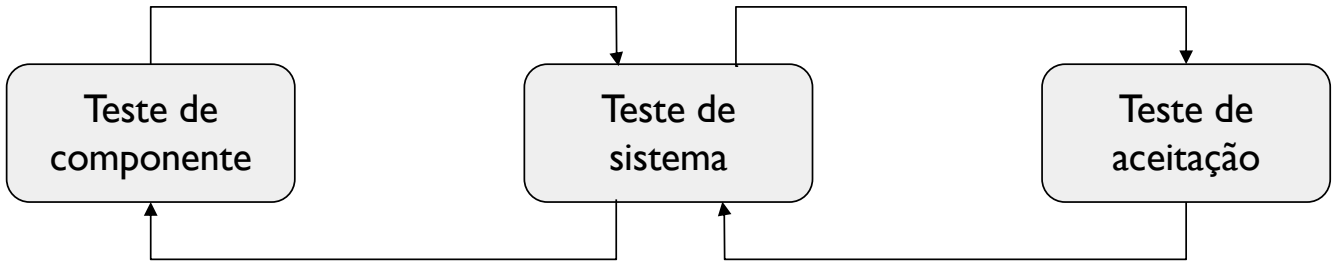
Processo de Software



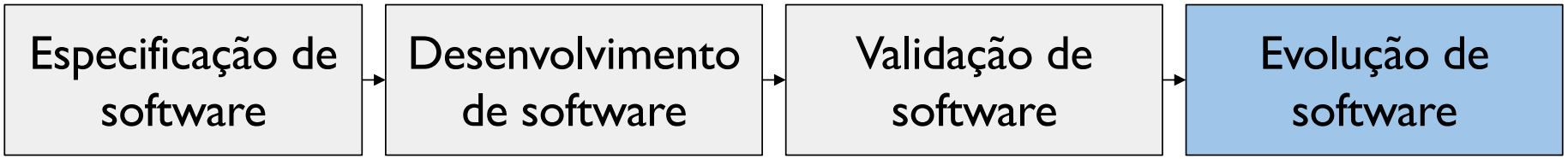
Validação de Software

- Validação de software ou Verificação e Validação (V&V) destina-se a **mostrar que o sistema está em conformidade** com a sua especificação.
- Verificações são **realizadas a cada estágio do processo** de software (p. ex., especificação de requisitos, projeto de sistema, código, etc.)
- O maior custo de validação incorre após a implementação, quando o sistema é operacional.

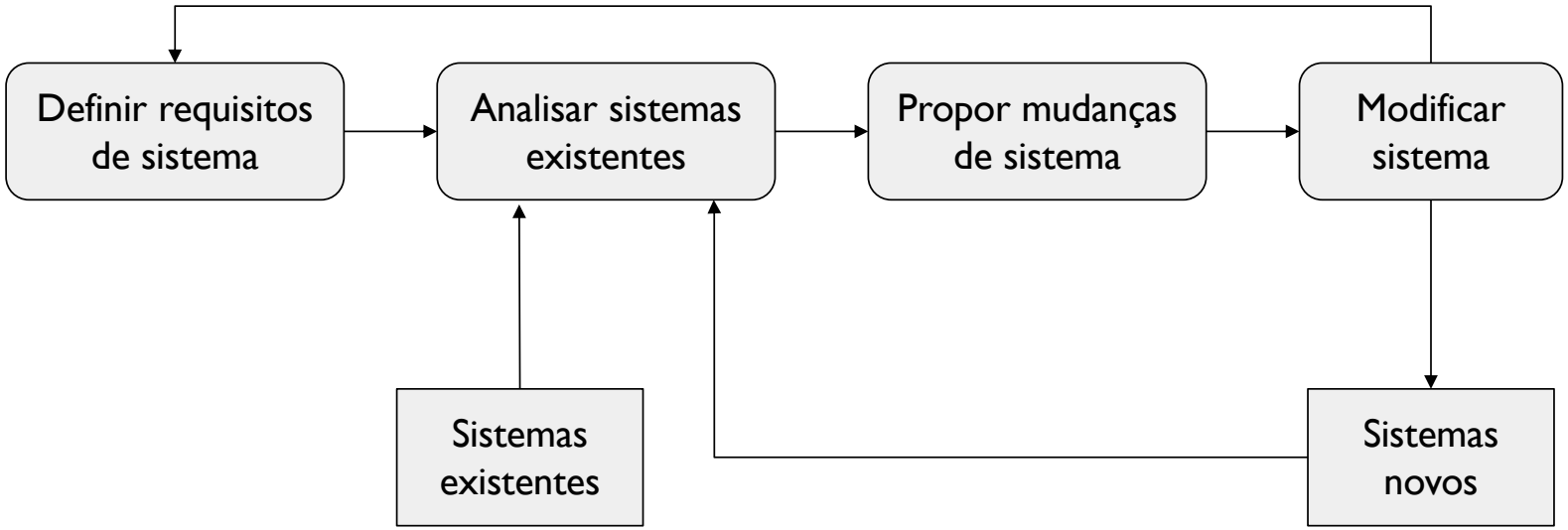
Validação de Software



Processo de Software



Evolução do Software



MODELOS DE PROCESSO DE SOFTWARE

Modelos de processo de software

- É uma **descrição simplificada** do processo e representa um processo sob **determinada perspectiva**.

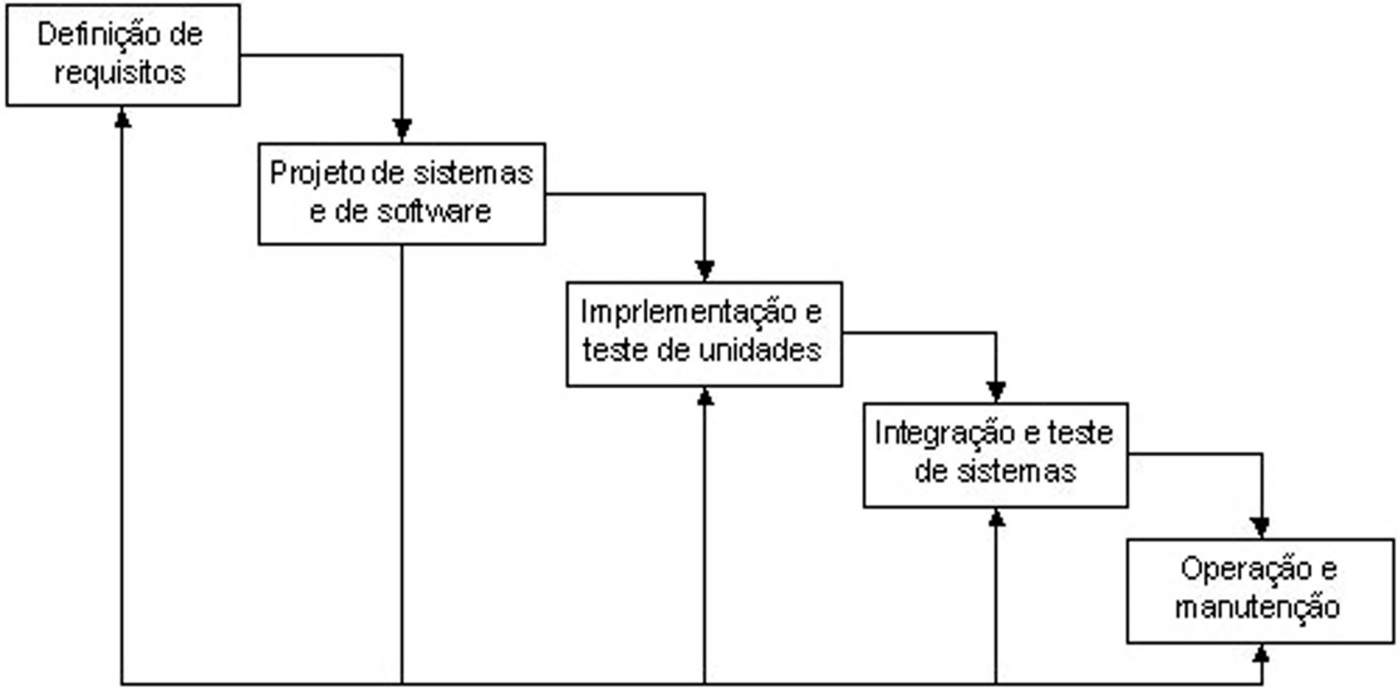
Modelo Cascata

- Primeiro modelo a **organizar** as atividades de desenvolvimento de software.
- Atividades:

(1) Análise e definição de requisitos,
(2) Projeto de sistema,
(3) Implementação e teste de unidade,
(4) Integração e teste de sistema,
(5) Operação e manutenção.

A fase seguinte não deve começar antes que a fase anterior tenha terminado.

Modelo Cascata



Modelo Cascata

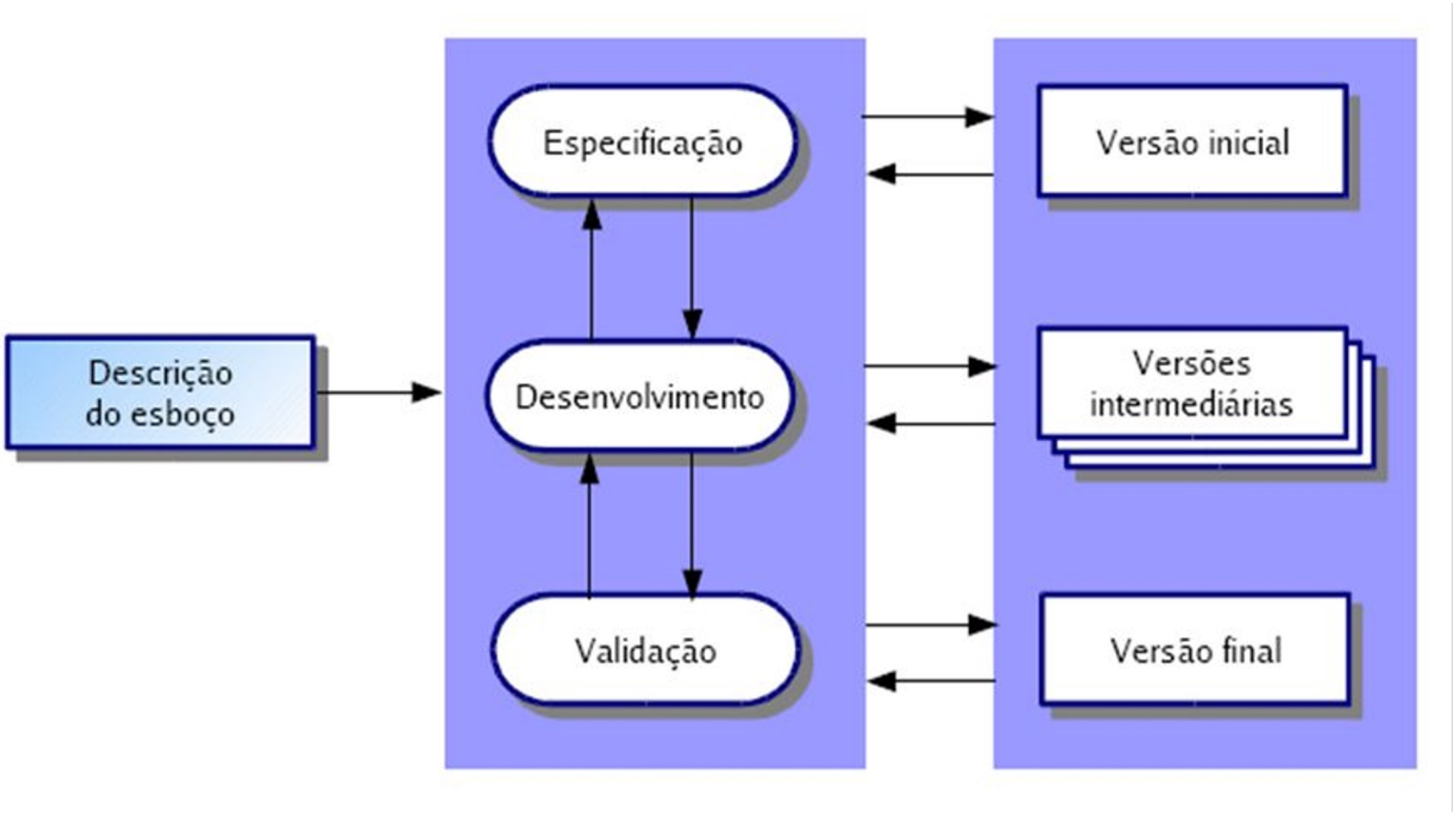
Desvantagens (problemas):

1. O resultado de cada fase envolve um ou mais documentos que são aprovados, gerando **muita documentação**.
2. A fase seguinte não deve iniciar até que a fase precedente tenha sido concluída.
3. Particionamento inflexível do projeto em estágios.
4. Apropriado somente quando os requisitos são bem compreendidos e as mudanças são raras.

Modelo Evolucionário

- Desenvolvimento de uma implementação inicial, exposição do resultado aos comentários do usuário e refinamento do resultado por meio de várias versões.
- As atividades de especificação, desenvolvimento e validação são intercaladas.

Modelo Evolucionário



Modelo Evolucionário

Vantagens:

- Os sistemas atendem às necessidades imediatas dos clientes.
- A especificação pode ser desenvolvida de forma incremental (usuários compreendem melhor o problema).

Modelo Evolucionário

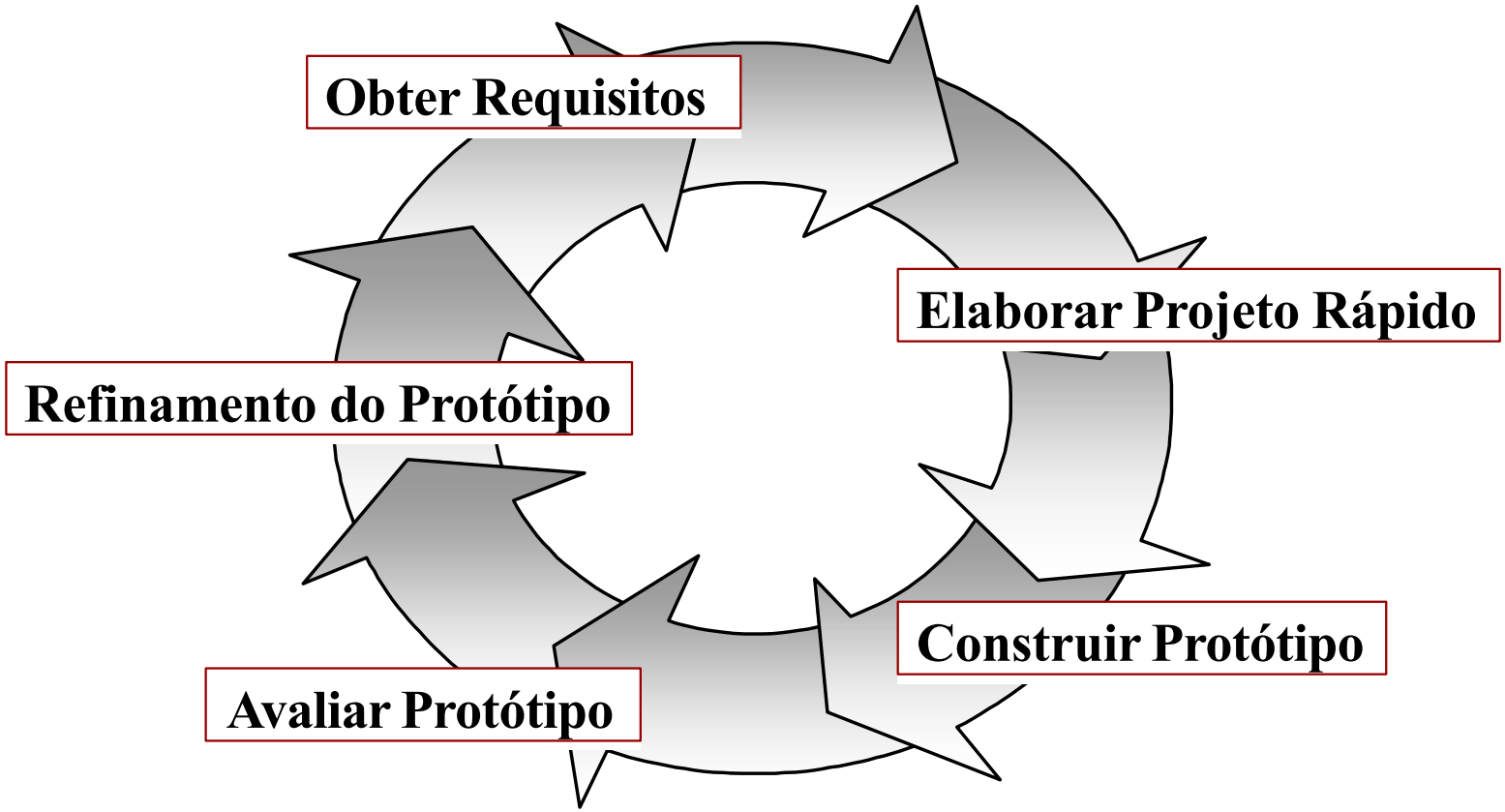
Desvantagens:

- O progresso é medido por meio dos produtos entregues.
- A mudança contínua tende a corromper a estrutura do software.
- A incorporação de mudanças torna-se cada vez mais difícil e onerosa.

O Modelo de Prototipação

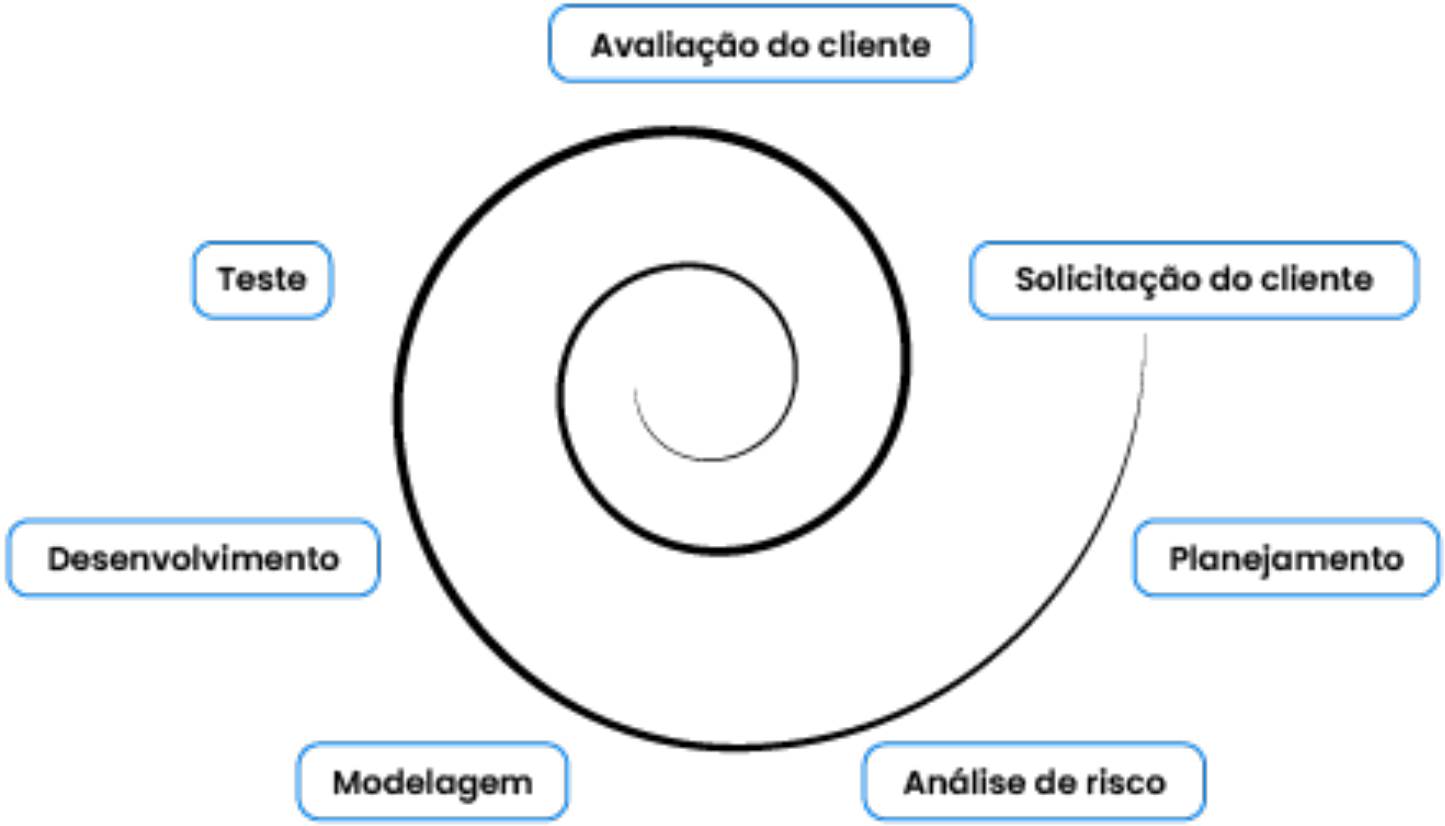
- O objetivo é entender os requisitos do usuário e, assim, obter uma melhor definição dos requisitos do sistema.
- Possibilita que o desenvolvedor crie um modelo (protótipo)do software que deve ser construído
- Apropriado para quando o cliente não definiu detalhadamente os requisitos.

O Paradigma de Prototipação



O Modelo Espiral

- O modelo espiral acopla a natureza iterativa da prototipação com os aspectos controlados e sistemáticos do modelo cascata.
- O modelo espiral é dividido em uma série de *atividades de trabalho* ou *regiões de tarefa*.


Avaliação do cliente
Solicitação do cliente
Planejamento
Análise de risco
Modelagem
Desenvolvimento
Teste

METODOLOGIA ÁGIL

- A metodologia ágil é uma abordagem de desenvolvimento de software que se baseia em princípios e valores que valorizam a colaboração, a flexibilidade, a entrega contínua de valor e a adaptação às mudanças

METODOLOGIA ÁGIL

- As metodologias ágeis têm como expectativa uma resposta mais rápida às necessidades dos clientes e às mudanças nos requisitos

PRINCÍPIOS

- **Colaboração e Comunicação:** As metodologias ágeis enfatizam a colaboração próxima entre os membros da equipe de desenvolvimento, bem como a comunicação regular com os clientes e partes interessadas. Isso ajuda a garantir que todos tenham uma compreensão clara dos objetivos e requisitos do projeto.

PRINCÍPIOS

- **Entrega Contínua de Valor:** Em vez de esperar até o final de um longo ciclo de desenvolvimento para entregar um produto, as metodologias ágeis promovem a entrega contínua de incrementos de valor para os clientes. Isso significa que partes utilizáveis do software são entregues em intervalos regulares.

PRINCÍPIOS

- **Flexibilidade e Adaptação:** As metodologias ágeis reconhecem que os requisitos e as prioridades podem mudar ao longo do tempo. Elas permitem que as equipes se adaptem a essas mudanças de forma eficaz, ajustando o trabalho conforme necessário.

PRINCÍPIOS

- **Iteração e Feedback:** As metodologias ágeis frequentemente usam ciclos curtos de desenvolvimento, chamados de "iterações" ou "sprints", nos quais uma parte do software é desenvolvida e depois revisada. Isso permite que a equipe receba feedback regular e faça melhorias contínuas.

PRINCÍPIOS

- **Pessoas mais que Processos e Ferramentas:** Embora processos e ferramentas sejam importantes, as metodologias ágeis valorizam mais as pessoas e suas interações. Acredita-se que equipes motivadas e colaborativas são fundamentais para o sucesso.

PRINCÍPIOS

- **Trabalho em Equipe Auto-organizada:** As equipes ágeis são frequentemente auto-organizadas, o que significa que têm um grau de autonomia para tomar decisões relacionadas ao projeto. Isso promove a responsabilidade e a motivação da equipe.

METODOLOGIA ÁGIL

- Frameworks e abordagens ágeis mais conhecidos incluem Scrum, Kanban, Extreme Programming (XP)

MANIFESTO ÁGIL

- https://agilemanifesto.org/iso/ptbr/manifesto.html

VAMOS PRATICAR

PASSOS DA ATIVIDADE

Definição do Produto:

1. Os participantes devem escolher um produto de software para implementar.
2. Pode ser uma aplicação web, um aplicativo móvel, um sistema de gerenciamento de tarefas, uma rede social simplificada, ou qualquer outra ideia de interesse do grupo

PASSOS DA ATIVIDADE

Identificação de Funcionalidades:

1. Os participantes devem listar as funcionalidades principais do produto
2. As funcionalidades podem incluir a criação de perfis de usuário, a visualização de conteúdo, a realização de ações específicas, como postar mensagens ou adicionar amigos, entre outras funcionalidades relevantes ao contexto do produto escolhido
3. As funcionalidades podem ser requisitos funcionais e não funcionais

PASSOS DA ATIVIDADE

Tomada de decisões:

1. Os participantes devem listar todas as escolhas feitas para definição do projeto com base no produto e nas funcionalidades principais do produto
2. Estas decisões abrangem:
    1. Seleção de tecnologias e frameworks
    2. Definição de estruturas de dados
    3. Algoritmos
    4. Padrões de projeto
    5. Componentes
3. Todas as decisões necessitam estar justificadas

REFERÊNCIAS

- Pressman, R.B. Software Engineering: A Practitioner'sApproach McGraw-Hill, Third Edition, New-York, EUA
- Rocha, A. R.C. and Maldonado,J.C. andWeber, K.C. Qualidade de Software: Teoria e PáticaPrentice-Hall 2001, SP, Brasil
- Cortes, M.L. and Chiossi, T.C.S. Modelos de Qualidade de Software Editora da Unicamp 2001, Campinas, SP, Brasil
- SEI-Carnegie Mellon University. The Capablity Maturity Model: Guidelines for Improving theSoftware ProcessAddison Wesley-USA
- Kan, H.S.Metrics and Modelsin Software Quality Engineering. Addison Wesley, 1995, USA

QUALIDADE DE SOFTWARE

João Choma Neto

joaochoma+aulas@gmail.com

https://github.com/JoaoChoma