

Principios SOLID

S → SRP (Single responsibility principle)

O → OCP (Open closed principle)

L → LSP (Liskov substitution principle)

I → ISP (Interface segregation principle)

D → DIP (Dependency inversion principle)

Single Responsibility Principle

Cada classe deve ser responsável por apenas uma parte da funcionalidade de um software e essa responsabilidade deve ser totalmente englobada por uma classe

Open Closed Design Principle

Classes, métodos e funções devem estar abertos para extensão e fechados para modificação

Liskov Substitution Principle

Objetos de uma superclasse devem ser substituíveis por objetos de uma subclasse sem dor break a aplicação

Um objeto de uma subclasse tem de se comportar de mesma forma e respeitar o input dos mesmos parâmetros

que os objetos da superclasse

Interface Segregation Principle

Um cliente não deve ser forçado a depender de uma interface que não usa

Dependency Inversion Principle

Em vez de definir as dependências estaticamente num componente deixamos o framework injetar as dependências dinamicamente

Requisitos

Parâmetros	Funcionais	Não Funcionais
O que são necessidade como detectar	Verbo obrigatório caso de uso	atributos não é obrigatório é um atributo qualitativo
resultado final	funcionalidade do produto	propriedade do produto
Detetor objetivo	facilmente detetável ajuda a verificar o funcionamento do software	dificilmente detetável ajuda a verificar a performance do software
Área de foco	necessidade do utilizador	expectativa do utilizador descreve como o produto funciona

Documentação	descreve o que o produto faz	
como testar	Teste funcional por exemplo no sistema, integração, End-to-end...	Teste não funcional por exemplo performance, stress, usabilidade
Teste de execução	feito antes do não funcional	feito depois do funcional
Info do produto	características do produto	propriedades do produto

Metodologias Tradicionais

Codificar e reparar (code-and-fix)

Cascata

Evolutiva

Transformações

Espiral

Codificar e Reparar

Desenvolvimento do processo em 2 passos:

Escrever código

Modificar código, de forma a reparar erros, melhorar ou acrescentar funcionalidade

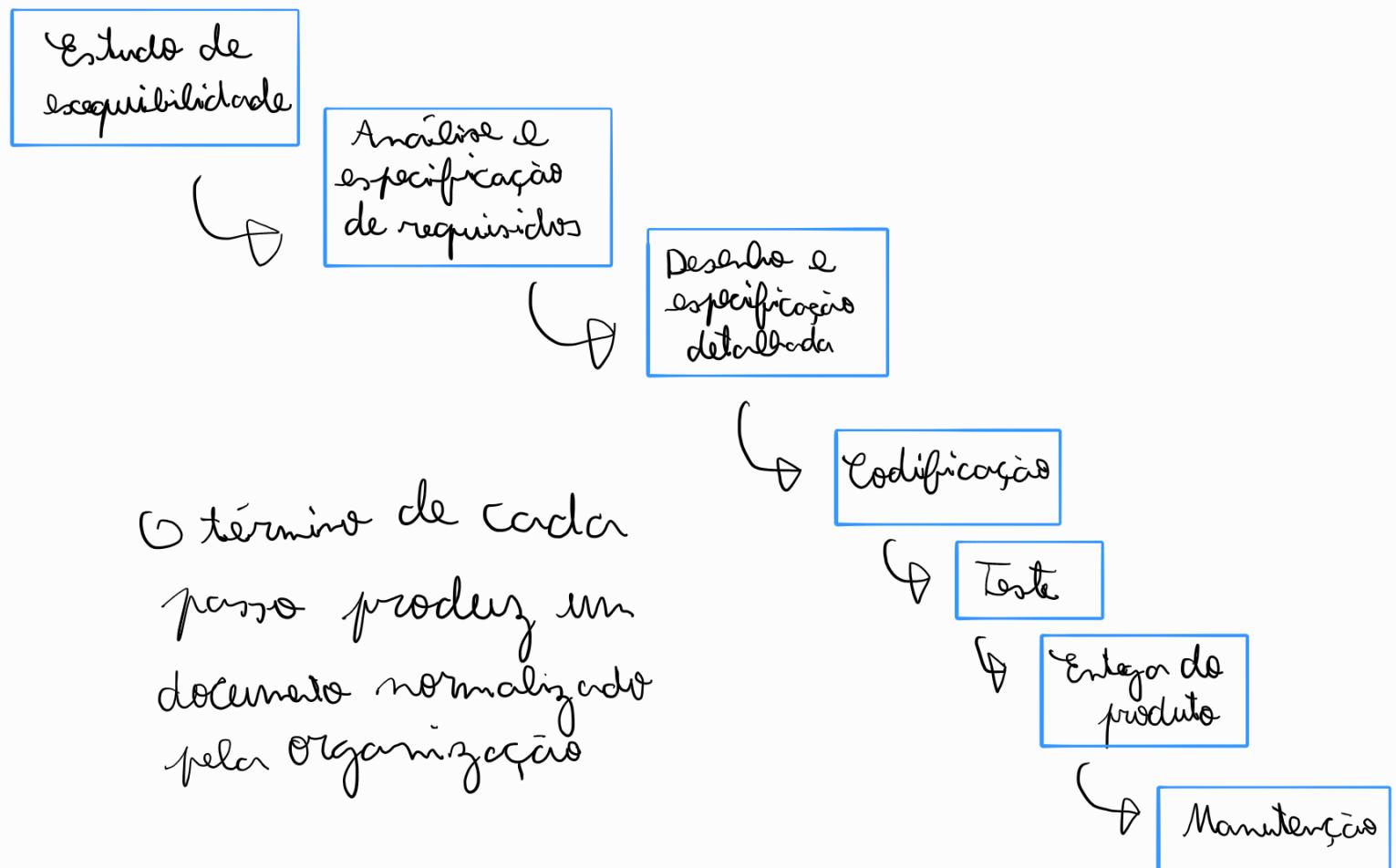
Faltas do modelo:

Dificuldade em gerir a complexidade de modo não estruturado

Falta de documentação

Impossibilidade de controlar o processo

Modelo em cascata



Estudo de exequibilidade: Serve para melhor os custos e os benefícios a desenvolver. Tipicamente magro

Análise e Especificação de Requisitos: Identifica as qualidades requeridas, sendo estas Funcionalidade, Desempenho e portabilidade. Deve seguir os princípios do software tendo separação de preocupações, abstração e modularização. Deve também ser feito em documento que especifique

Desenho e Especificações Detalhadas: Decomposição do sistema em duas partes, a lógica (desenho de alto nível) e a física (definição de estruturas de dados e algoritmos).

Codificação: Escrever os softwares numa linguagem de programação segundo os padrões de finidas pela empresa.

Teste → Teste individual e de integração dos diversos módulos.

Após isso é realizado um teste global.

Entrega → Em duas fases: Beta → Entrega seguindo condições controladas
Oficial → Distribuída sem restrições

Mantenção → geralmente é 20% corretiva, 20% adaptativa e 50% evolutiva

É um desenvolvimento disciplinado, planeado e gerido. Devendo a produção começar apenas quando os objetivos são completamente conhecidos.

Lançar com Retorno

Permite voltar o trâns para corrigir

Análise Crítica

A análise de requisitos só é efectuada quando já se realizou a entrega do produto. difícil de adaptar a mudanças

Metodologia Evolutionária

- Software é desenvolvido em partes (incrementalmente)
- Envolve o utilizador em todas as fases
- Usa protótipos para testar ideias rapidamente
- Vantagens: flexibilidade, feedback contínuo, adapta-se melhor a mudanças.

Protótipos

- Protótipos são versões simples para testar ideias
- Deverem ser rápidos e fáceis de construir
- Ajuda a capturar os requisitos reais do utilizador

Modelo em Espiral

Focus-se na análise e gestão de riscos

Mistura elementos de outras metodologias.

E

