FaculdadeBarretos
www.faculdadebarretos.com.br

Projetos de *Software*

(aula 2)

O Processo de *Software*

Me. Diogo Tavares da Silva
contato: *tavareko@gmail.com*

O que é o processo de software?

- Conjunto coerente de atividades que leva à produção de um produto de *software*
  - Pode ser chamado também de ciclo de vida de *software*

Processo de *software*

- Não existe um único tipo de processo de software
  - Adaptado às necessidades
    - tipo de *software*
    - padrão pré-estabelecido pela empresa
    - base de conhecimento dos analistas envolvidos
    - ...
  - No entanto, existem atividades básicas que são fundamentais em qualquer processo de *software*

Atividades básicas do processo de *software*

- Especificação de *software*
  - O que estamos desenvolvendo?
- Desenvolvimento de *software*
  - projeto e implementação do sistema
- Validação do *software*
  - verificação de funcionamento e atendimento das necessidades do cliente
- Evolução do *software*
  - Implantação, manutenção e verificação da necessidade de mudanças

Especificação de *software*

- Fase também conhecida como **engenharia de requisitos**
- Compreender o que o sistema de fazer, de que forma deve ser feito e quais são suas restrições de operação
- Etapa **FUNDAMENTAL**:
  - Não compreensão dos requisitos do sistema acarreta em problemas futuros no desenvolvimento do projeto

Especificação de *software*

- Dividida em quatro atividades principais:
  - **Estudo de viabilidade**
    - Verifica-se se o desenvolvimento do *software* é viável considerando-se fatores econômicos, ambientais e tecnológicos
      - O *software* vai de fato resolver as necessidades do cliente?
      - As tecnologias de *hardware* e *software* existentes permitem o desenvolvimento?
      - é economicamente viável desenvolver o software com os recursos disponíveis?

Especificação de *software*

- **Levantamento e análise de requisitos**
  - Processo de obtenção dos requisitos do sistema
  - Pode-se capturar os requisitos por meio de diversas técnicas:
    - Entrevistas com usuários e clientes;
    - Questionários
    - Reuniões
    - Análise dos processos de trabalho

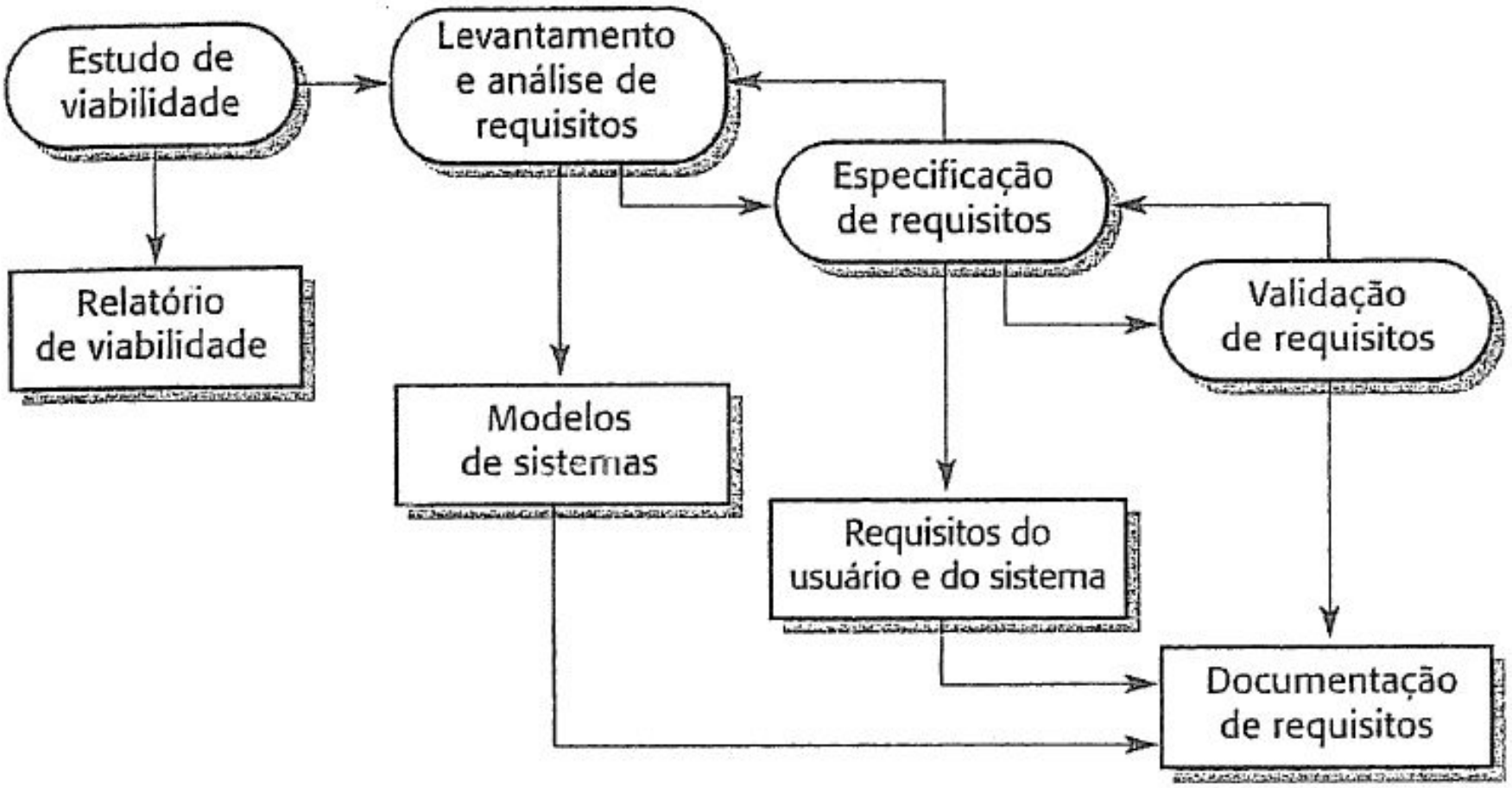
Especificação de *software*

- **Especificação dos requisitos**
  - Busca descrever os requisitos coletados formalmente por meio de dois documentos:
  - **Documento de Requisitos de Usuário**
    - Em que se descrevem os requisitos de maneira mais geral e abrangente
    - O usuário deve compreender o documento
  - **Documento de Requisitos do Sistema**
    - Descrição mais técnica dos requisitos, como maior detalhamento de aspectos de implementação

Especificação de *software*

- **Validação de requisitos**
  - Verificação dos requisitos quanto sua pertinência, consistência e integralidade.
    - Envolvimento dos clientes stakeholders na verificação
    - Durante esse processo são descobertos erros na documentação de requisitos.

Especificação de *software*



Projeto e implementação de *software*

- Etapa de projeto que busca transformar uma especificação de sistema em um sistema executável.
- Envolve duas atividades principais:
  - **Projeto do sistema**
    - Modelagem do sistema de acordo com a especificação coletada
  - **Implementação do sistema**
    - Codificação dos componentes do sistema de acordo com a modelagem projetada

Projeto do *software*

- Descrição da estrutura do *software* a ser implementada posteriormente
  - Modelam-se:
    - Os componentes que integram o sistema (subsistemas)
    - Os dados que são manipulados pelo sistema
    - As interfaces entre os componentes do sistema
    - Os algoritmos utilizados.

Projeto do *software*

- As atividades específicas da etapa de projeto de software são:
  - **Projeto de arquitetura**
    - Os subsistemas que constituem o sistema e suas relações são identificados e documentados.
  - **Especificação abstrata**
    - Para cada subsistema, é produzida uma especificação abstrata suas funções e das restrições dentro das quais deve operar.

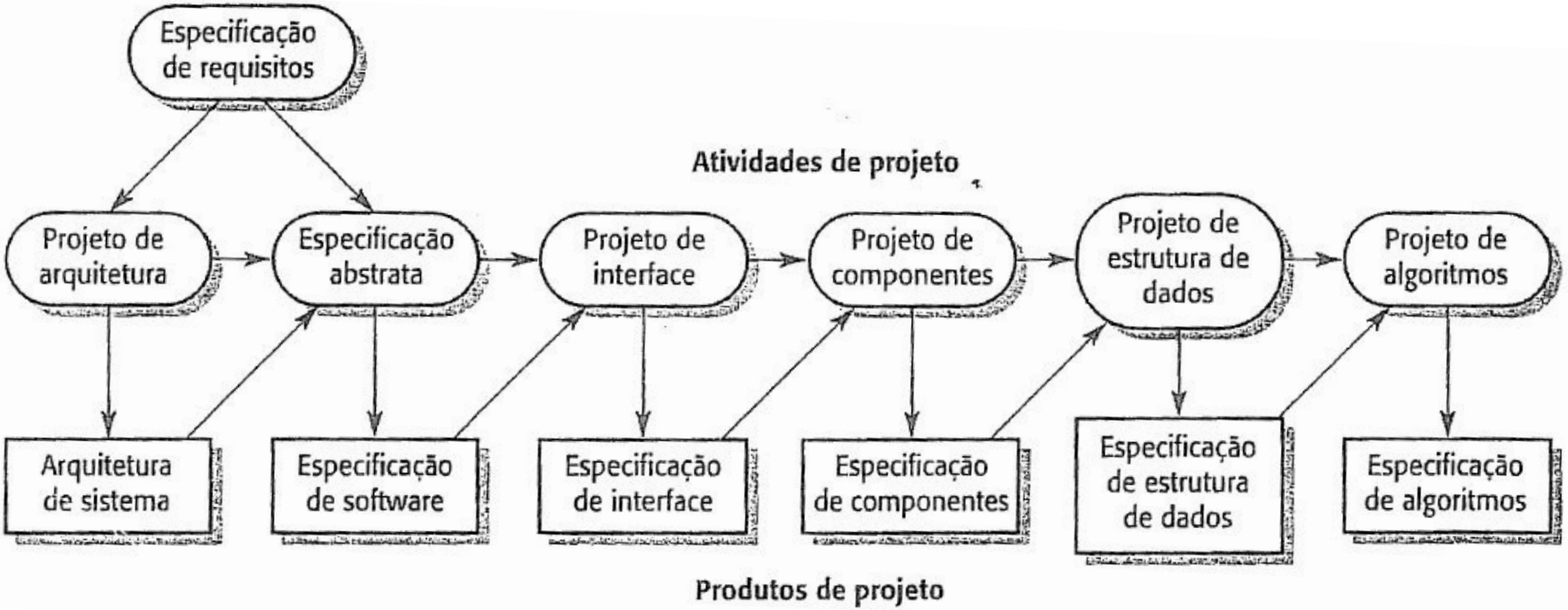
Projeto do *software*

- **Projeto de interface**
  - As interfaces com outros subsistemas são projetadas e documentadas.
    - A especificação de interface não pode ser ambígua
      - definir claramente os padrões de dados e serviços providos e recebidos por um subsistema.
      - Prover baixo acoplamento

Projeto do *software*

- **Projeto de componentes**
  - Funcionalidades são agrupadas em diferentes componentes e as interfaces entre componentes são projetadas
- **Projeto de estrutura de dados**
  - Projeto detalhado das estruturas de dados utilizadas na implementação do sistema
- **Projeto de algoritmos**
  - Os algoritmos utilizados para desenvolver as funcionalidades do programa são projetados e especificados

Projeto do *software*



Metodologias de análise de projeto

- Conjuntos de modelos de processo de projeto, notações, representações gráficas, formulários regras e diretrizes de projeto de sistema
- Atualmente existem duas abordagens mais populares:
  - **Análise estruturada**
    - DFD, DTE, DD e DER
  - **Análise orientada a objeto**
    - Modelagem UML

Programação e depuração de *software*

- O processo de programação de *software* é algo pessoal
  - Varia de um desenvolvedor para outro
  - Necessidade de padrões e boas práticas de implementação
- Dentre os aspectos mais importantes durante o processo de programação destacam-se a **documentação de código** e **depuração de erros.**

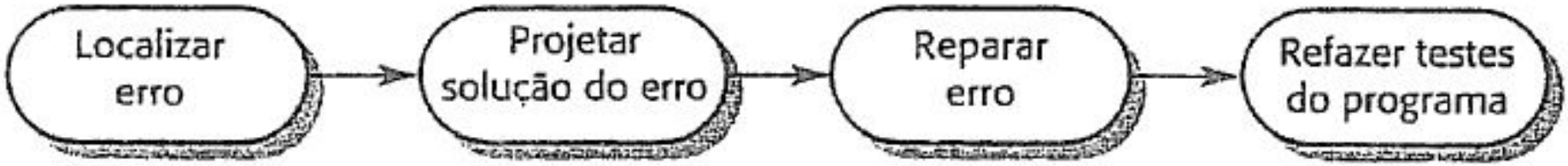
Documentação de código fonte

- Inserção de comentários e aspectos necessários para a compreensão da implementação do *software*
  - deixar claro o que realizam trechos de códigos, como saltos condicionais e laços de repetição
  - definir do que se tratam as classes, objetos e variáveis utilizadas e definir bem as funções, descrevendo suas entradas e retornos.
- Sempre que houver suporte, utilizar geradores de documentação padrão da linguagem (ex: *javadocs*)

Depuração de código

- Realização de testes de execução do código que está se desenvolvendo para identificar possíveis erros
- Consiste em localizar e sanar os erros de sistema durante o processo de programação
  - Depuração Manual:
    - Inserir variáveis de verificação e *"prints"* de status manualmente
  - Ferramentas de *debugging*
    - Depuradores independentes (ex: dbg) ou próprios de IDEs

Depuração de código



Verificação e validação de *software*

- Verificar se o sistema desenvolvido está de acordo com sua especificação e atende às expectativas do cliente comprador
- Verificação de conformidade das etapas de desenvolvimento de software (requisitos, projeto e implementação)

Verificação e validação de *software*

- Validação e testes realizados de modo estruturado:
  - Testes são realizados de forma sistemática nos subsistemas, componentes e estruturas que compõem o sistema
  - abordagem ***bottom-up:***
    - Descobrir e resolver defeitos em menor escala primeiro para reduzir a incidência de erros em maior escala.

Estágios do processo de testes

- Atividades de teste:
  - **Teste de unidade**
    - Componentes individuais são testados de forma independente para garantir que operem corretamente.
  - **Teste de módulo**
    - Componentes relacionados são agrupados em módulos que são testados juntamente para verificar a conformidade de seu funcionamento.

Estágios do processo de testes

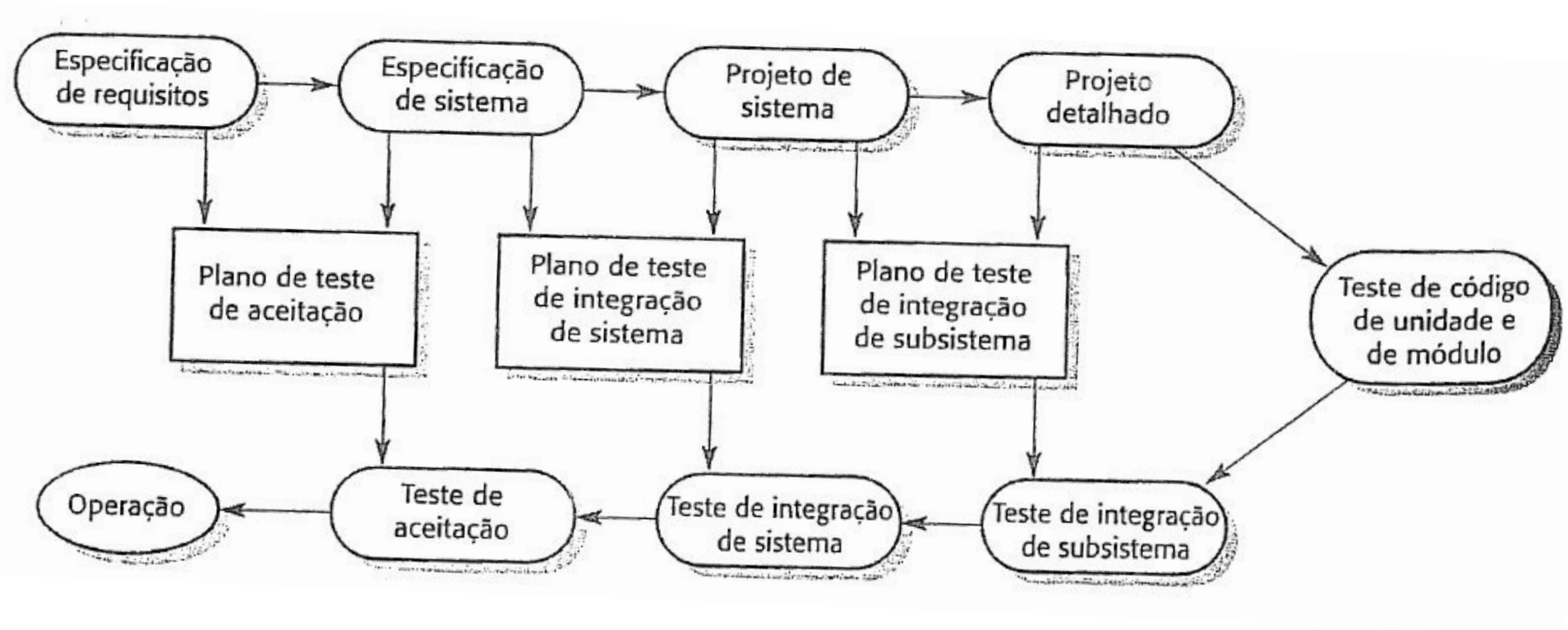
- **Teste de subsistema**
  - Testa-se conjuntos de módulos integrados em subsistemas.
    - Problema recorrente: discordância entre interfaces de dados.
      - Interfaces devem ser bem definidas
      - Módulo está recebendo e enviando os dados nos formatos corretos para módulos relacionados?

Estágios do processo de testes

- **Teste de sistema**
  - Integração total dos subsistemas (*build* completa).
    - Encontrar erros resultantes de interações não previstas entre subsistemas e problemas recorrentes de interface de dados.
    - validar o sistema em relação a seus requisitos funcionais e não funcionais.
      - funcionalidades e características de execução

Estágios do processo de testes

- **Teste de aceitação**
  - Estágio final do processo de testes
    - Uso de dados fornecidos pelo cliente ao invés de dados simulados em ambiente de desenvolvimento (**teste *alfa***)
    - Uso limitado do sistema para usuários específicos em ambiente de produção (**teste *beta***)
  - Pode revelar erros e omissões na definição de requisitos do sistema



Evolução de *software*

- *Software* é flexível
  - mudanças podem ser feitas com “facilidade”
    - mesmo caras, modificações no *software* são mais baratos que a alteração de um projeto de *hardware*

Evolução de *software*

- **Manutenção do sistema**
  - Realizar alterações em um produto de *software* depois dele entrar em operação devido a mudanças de requisitos.
  - *Softwares* podem mudar com frequência
    - Lembrem-se da aula anterior!

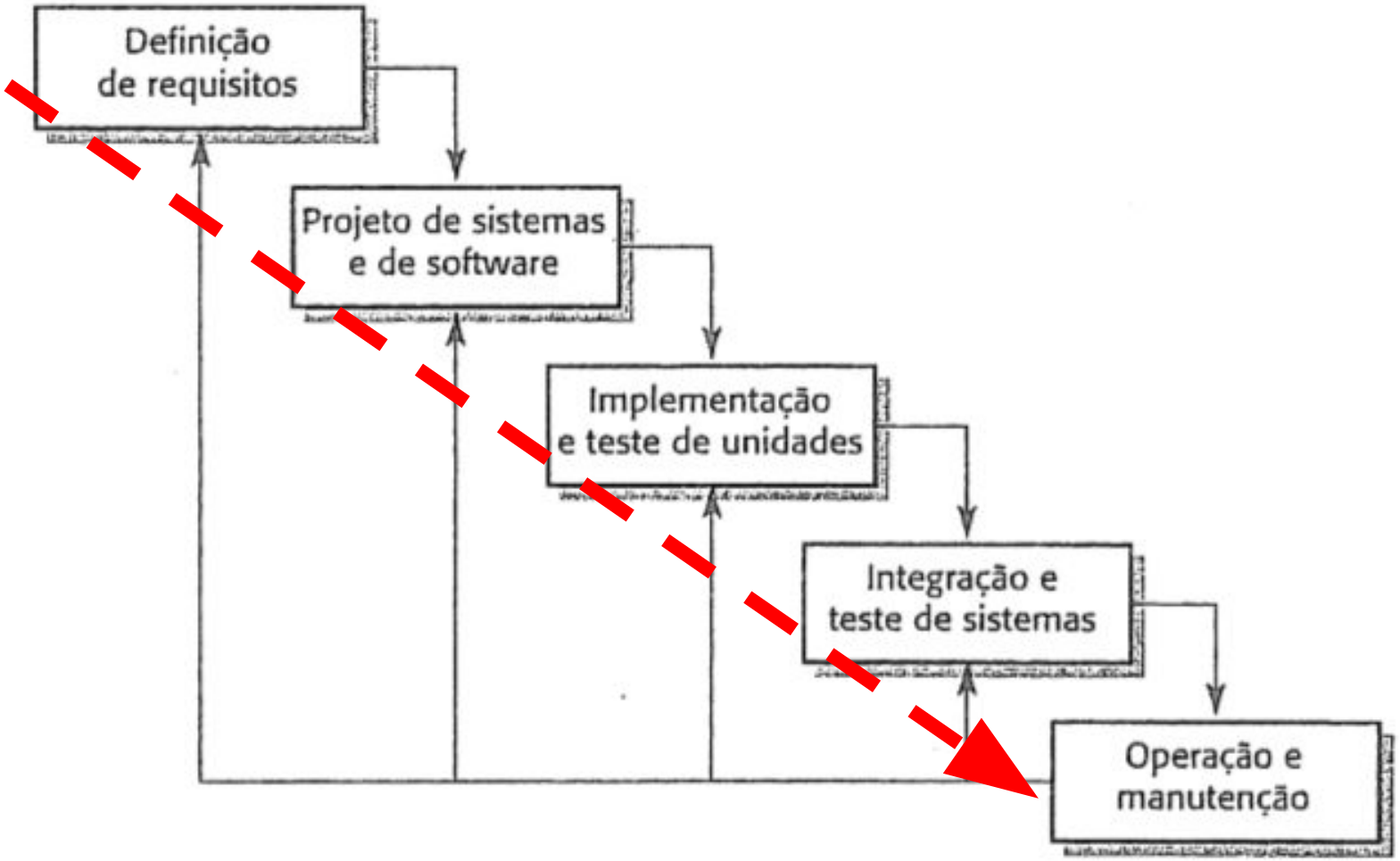
Modelos de processo de software

- Como discutimos anteriormente, o ciclo de vida de *software* não é necessariamente sequencial.
  - Diferentes tipos de produtos de *software*
    - diferentes tipos de abordagens do processo de software
- **Modelos de processo (Paradigmas de processo)**
  - Descrição genérica de uma classe de abordagens de processos de *software* em comum

Modelo em “cascata” (*waterfall*)

- Processos tradicionais de engenharia
- Fases bem definidas que ocorrem sistematicamente uma após a outra
- Uma fase se inicia somente quando todos os resultados produzidos pela fase anterior são aceitos
  - **Desvantagens**
    - Alterações de requisitos são difíceis de lidar
    - Grande volume de retrabalho

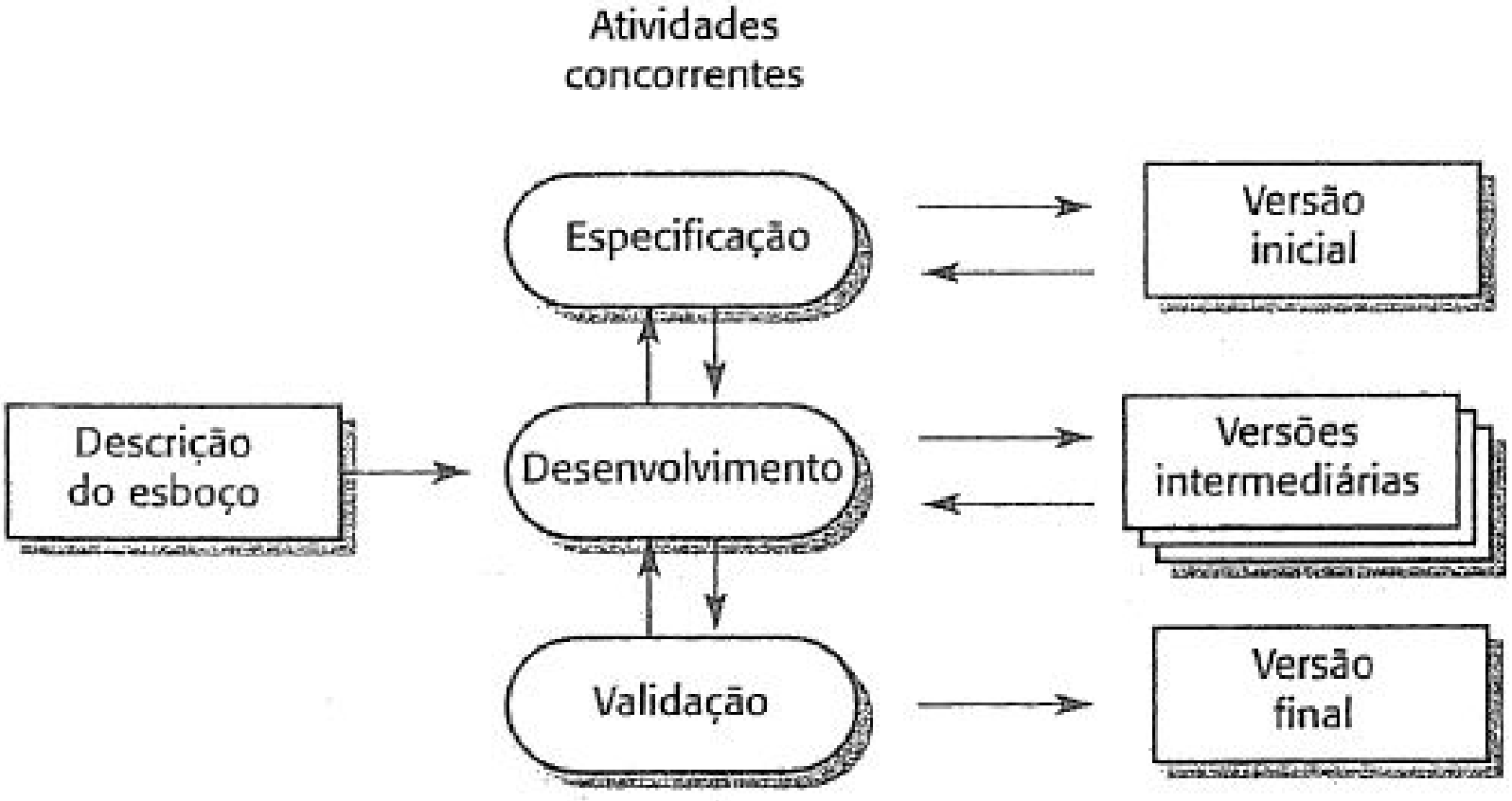
Modelo em “cascata” (*waterfall*)



Modelo evolucionário

- Desenvolver uma versão inicial
- pedir o retorno dos clientes sobre o sistema
- desenvolver versões intermediárias que reflitam nas necessidades imediatas dos clientes
- Desenvolver várias versões até que um sistema finalizado seja aceito pelo cliente
- **Desvantagens:**
  - Difícil de medir o progresso do projeto
  - Imediatismo
    - requisitos mal definidos

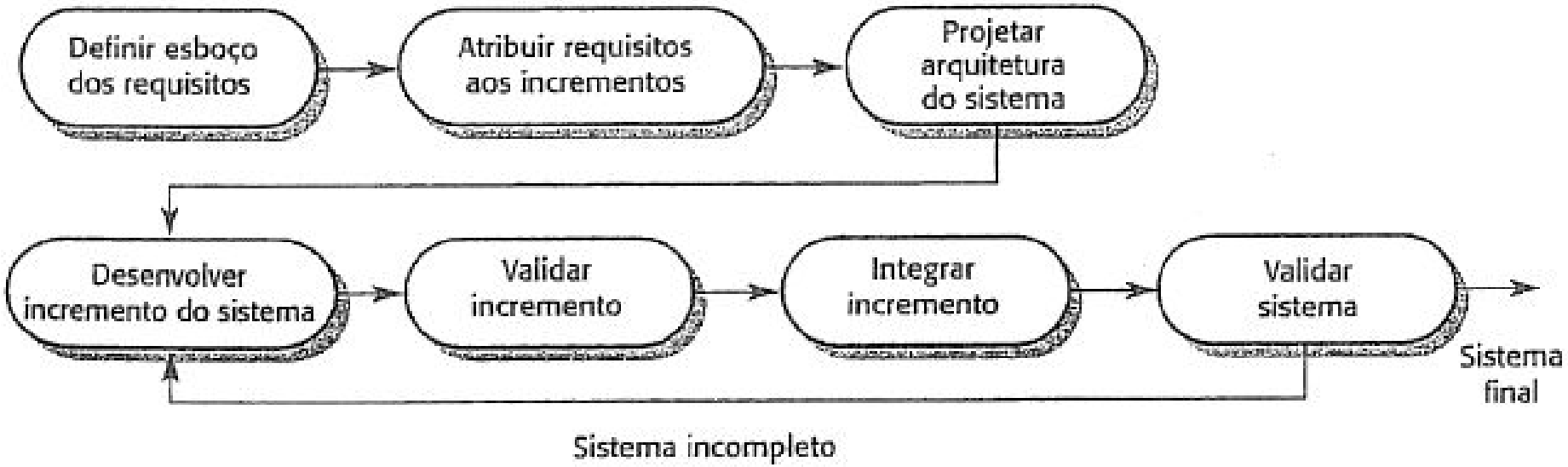
Modelo evolucionário



Modelo Incremental

- Combinar vantagens dos métodos cascata e evolucionário
- *software* desenvolvido em incrementos
- Disponibilizar versão mais generalizada do sistema para o usuário após a primeira iteração
  - Refinamento dos requisitos
    - desenvolver incrementos usando processo em cascata ou evolucionário, de acordo com a clareza como novos requisitos são refinados

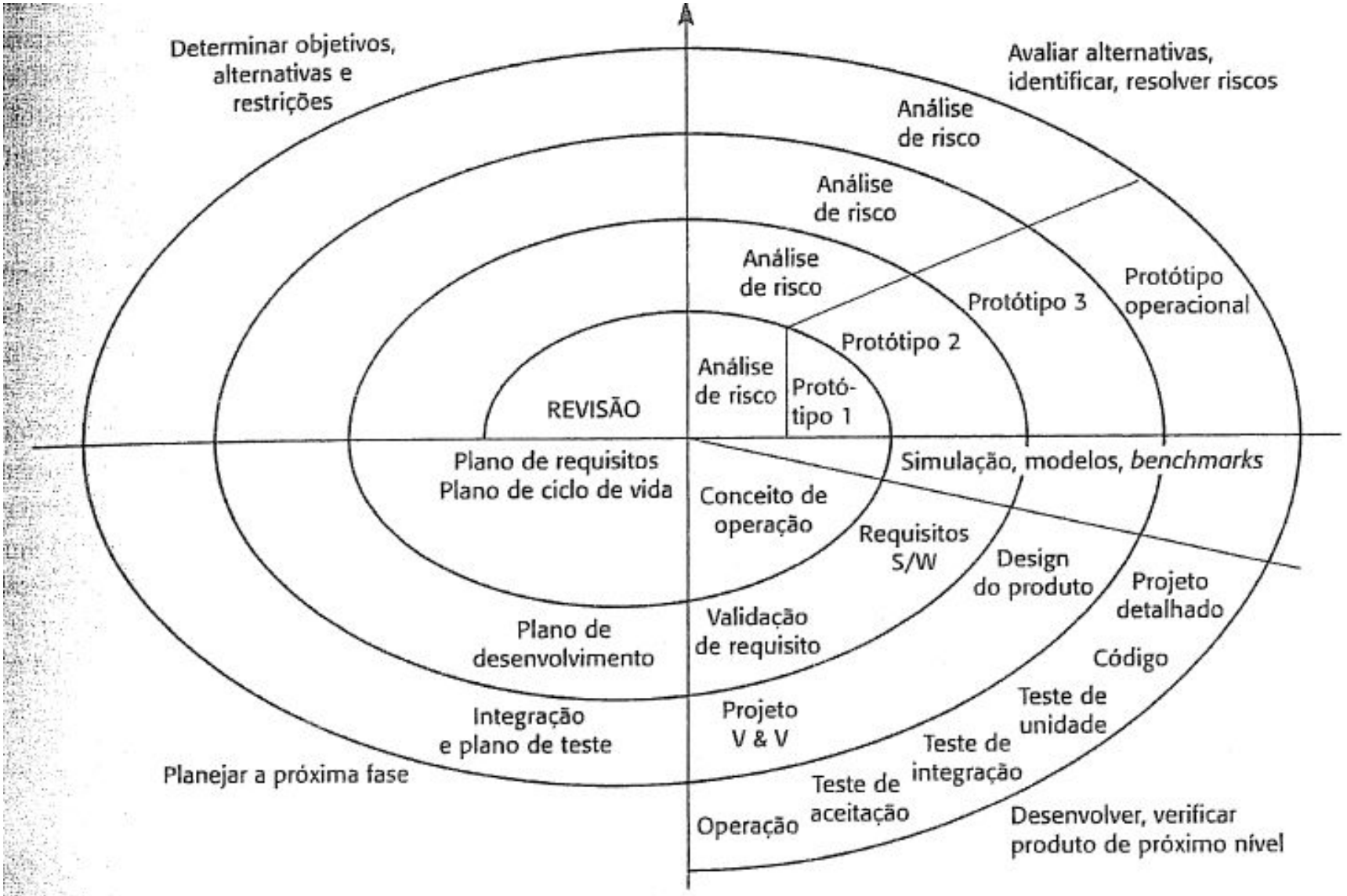
Modelo incremental



Modelo em espiral

- Desenvolvimento é visto como uma espiral que alterna entre as fases do processo.
- Cada *loop* representa uma fase do processo
  - *loops* internos representam fases de análise de requisitos e projetos
  - Loops mais externos representam implementações e testes

Modelo em espiral



Reúso de software

- **Abordagem informal**
  - Aproveitamento de códigos que realizam tarefas similares às exigidas
    - aplicar modificações e incorporar no projeto
- **Abordagem formal**
  - Desenvolvimento e uso de componentes reutilizáveis (Padrões de projetos, Conceitos de orientação a objetos).
  - **Outra abordagens:**
    - Incorporação de componentes de terceiros
    - Uso de arquitetura orientada a serviços (SOA).

Desenvolvimento com reúso de *software*