Modelo de Processos

UNIP - Araraquara

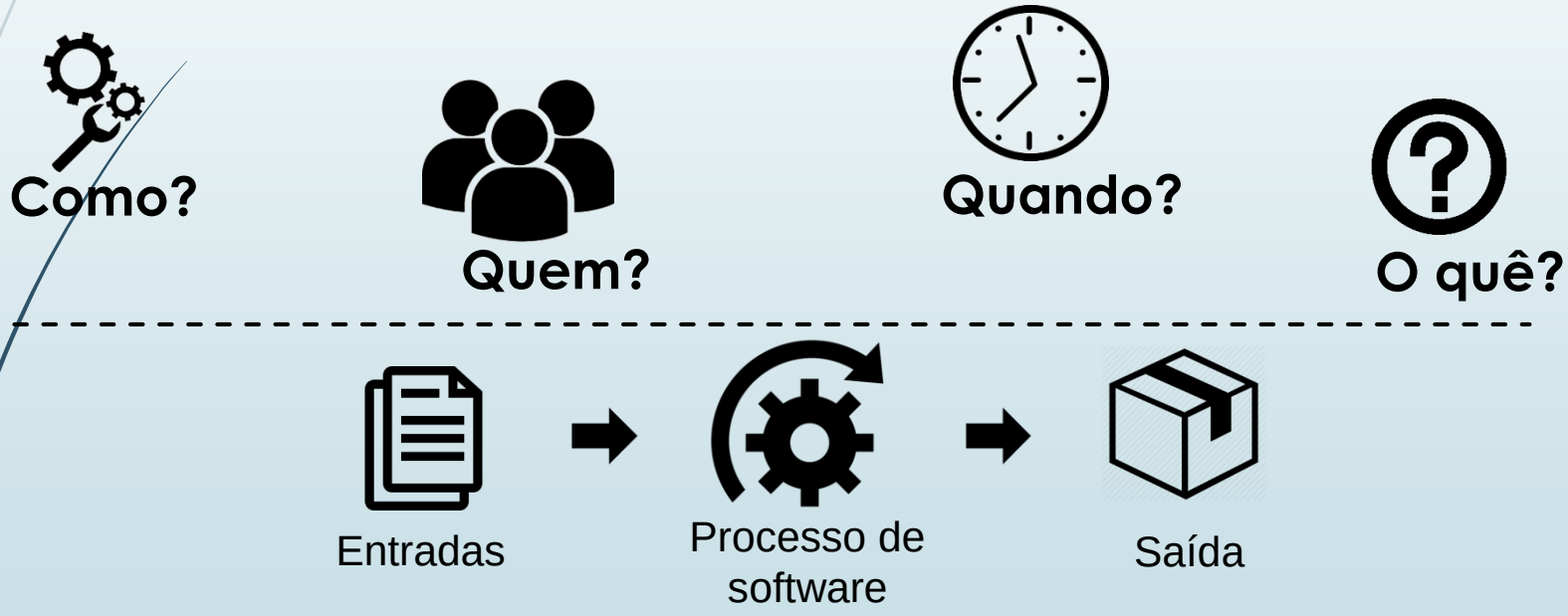
Curso: Ciência da Computação
Disciplina: Engenharia de Software
Profº: João Paulo Moreira dos Santos

Conteúdo

- Processo de Software
- Modelos de Processo
- Comparação: cascata, evolucionário e RUP
- Ferramentas Case

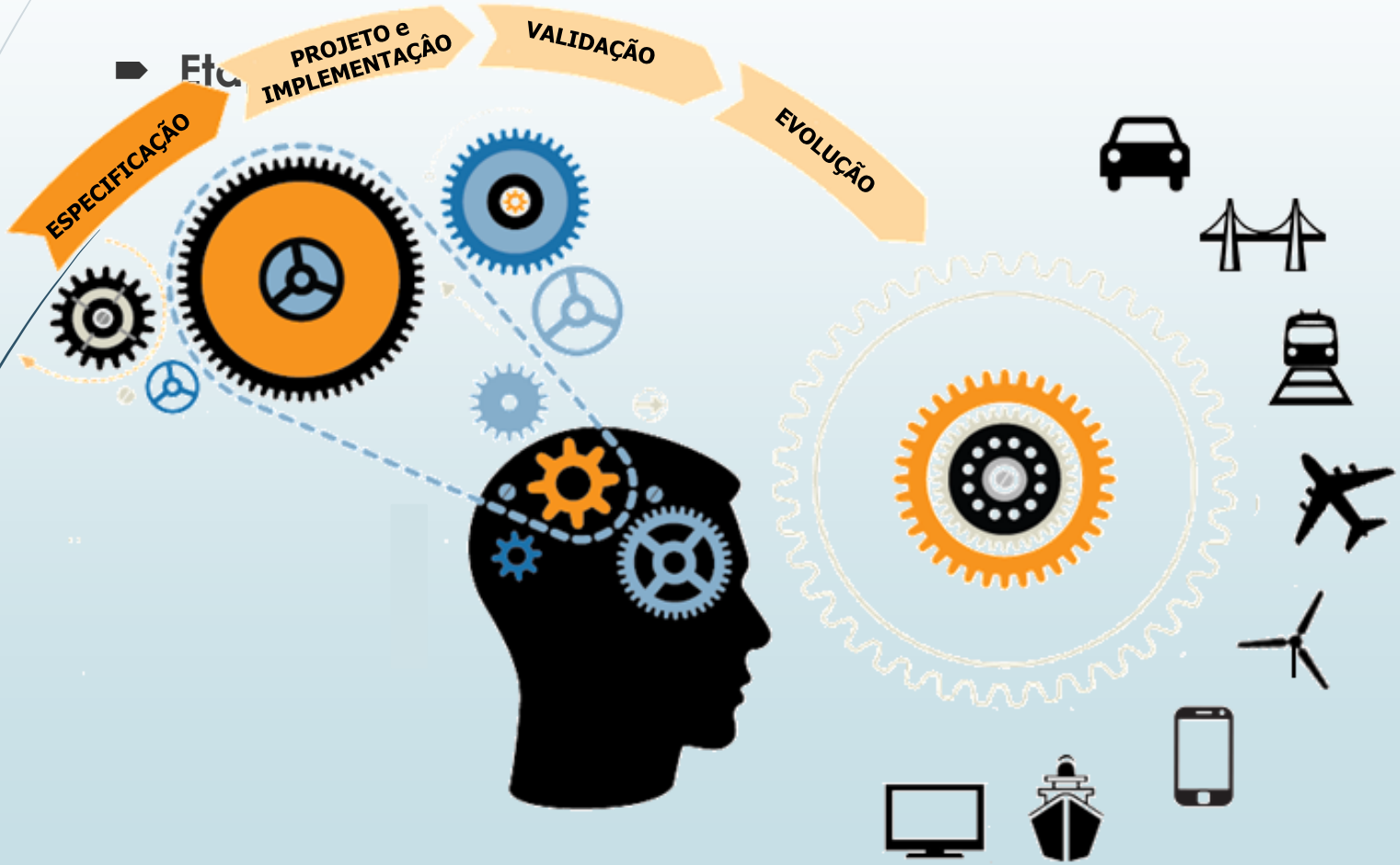
Processo de Software

- Um conjunto estruturado de **atividades**, **procedimentos**, **artefatos** e **ferramentas** necessários para o desenvolvimento de um sistema de software



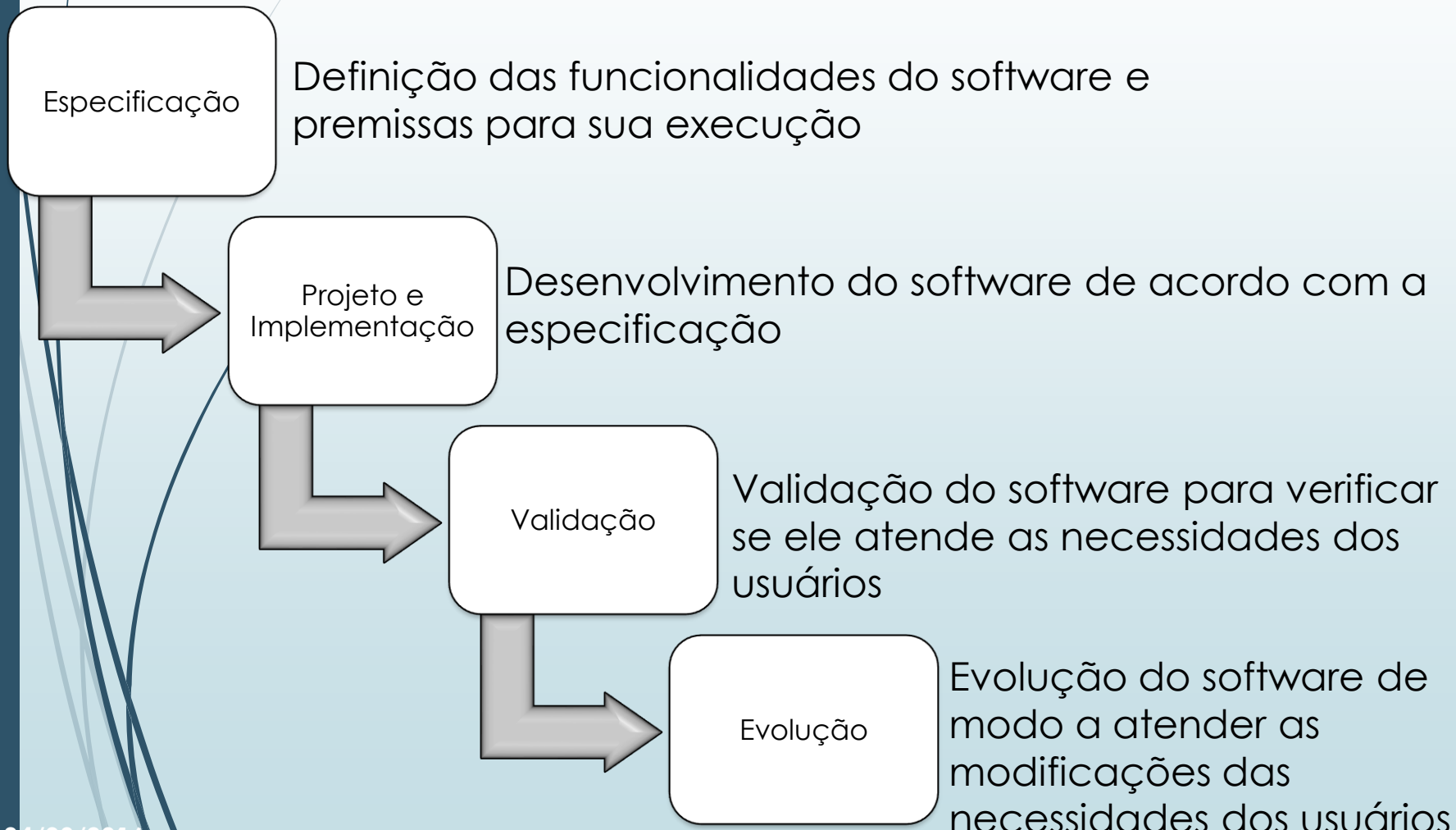
Não existe um processo ideal

Processo de Software



Processo de Software

- **Etapas:**



Processo de Software - Objetivos

- Visa a assegurar o desenvolvimento de software:
  - com prazos e necessidade de recursos definidos;
  - com elevada produtividade (de forma econômica);
  - com qualidade assegurada;
- Permite:
  - organizar
  - instrumentar
  - planejar
  - acompanhar projetos
  - treinar equipes

Modelos de Processo de software

- Apresenta a **descrição de um processo** de uma perspectiva particular, normalmente focando apenas em **algumas atividades**
- São utilizados para explicar diferentes abordagens do desenvolvimento de software
- Torna-se essencial para a definição de processos eficientes, capazes de serem replicados

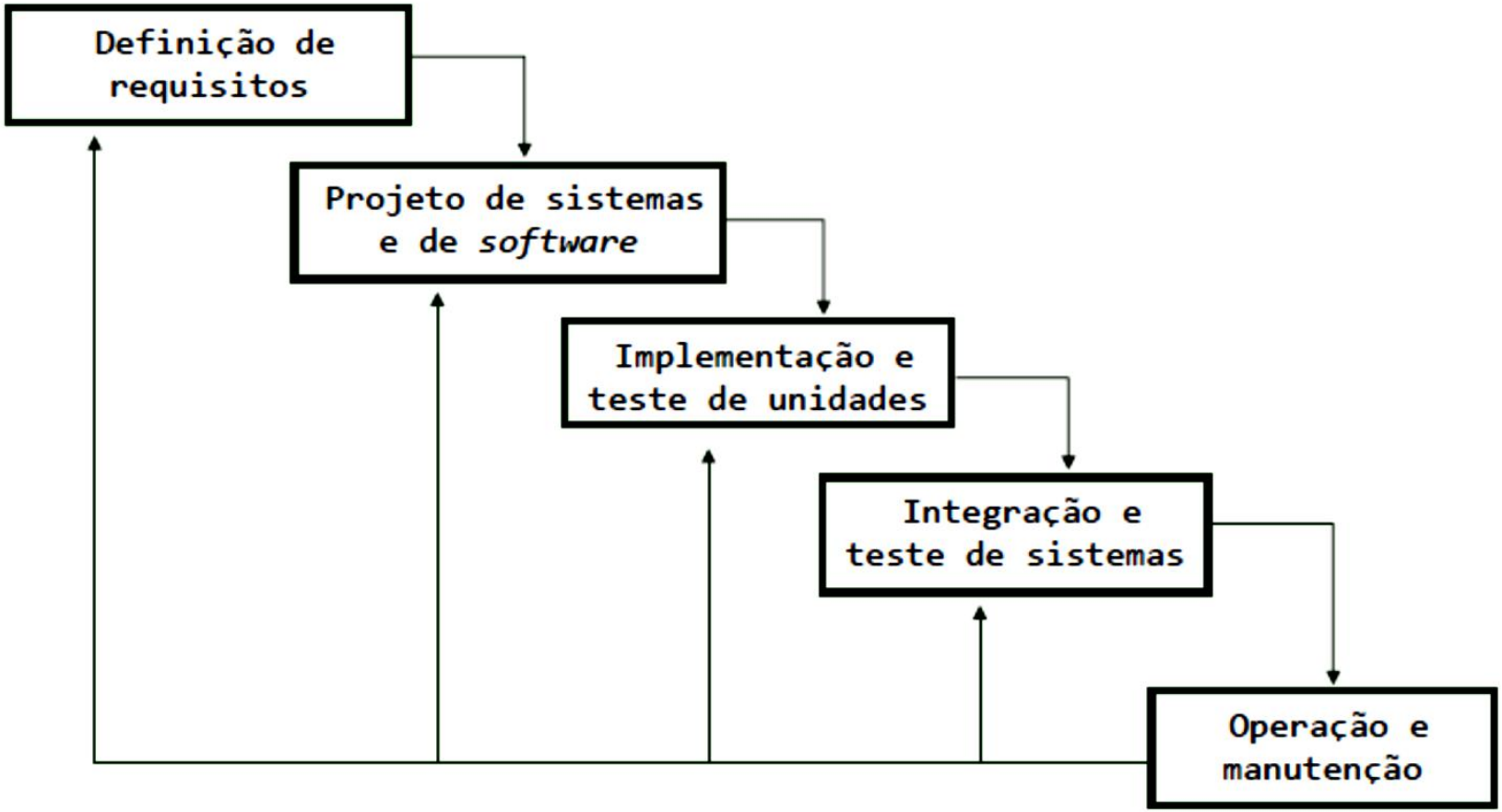
Modelos de Processo de software

- Modelos Genéricos
  - Modelo em Cascata
  - Modelo Evolucionário
  - Engenharia de Software Baseada em Componentes
- Prototipação
- Processo Iterativo
  - Incremental
  - Espiral

Modelo Cascata

- Primeiro modelo a **organizar** as atividades de desenvolvimento.
- Fases
  - Definição e análise de requisitos
  - Projeto do sistema e do software
  - Implementação e testes de unidade
  - Integração e testes do sistema
  - Operação e manutenção
- Uma fase tem de estar completa antes de passar para a próxima.
  - Saídas das fases são acordadas contratualmente

Modelo Cascata



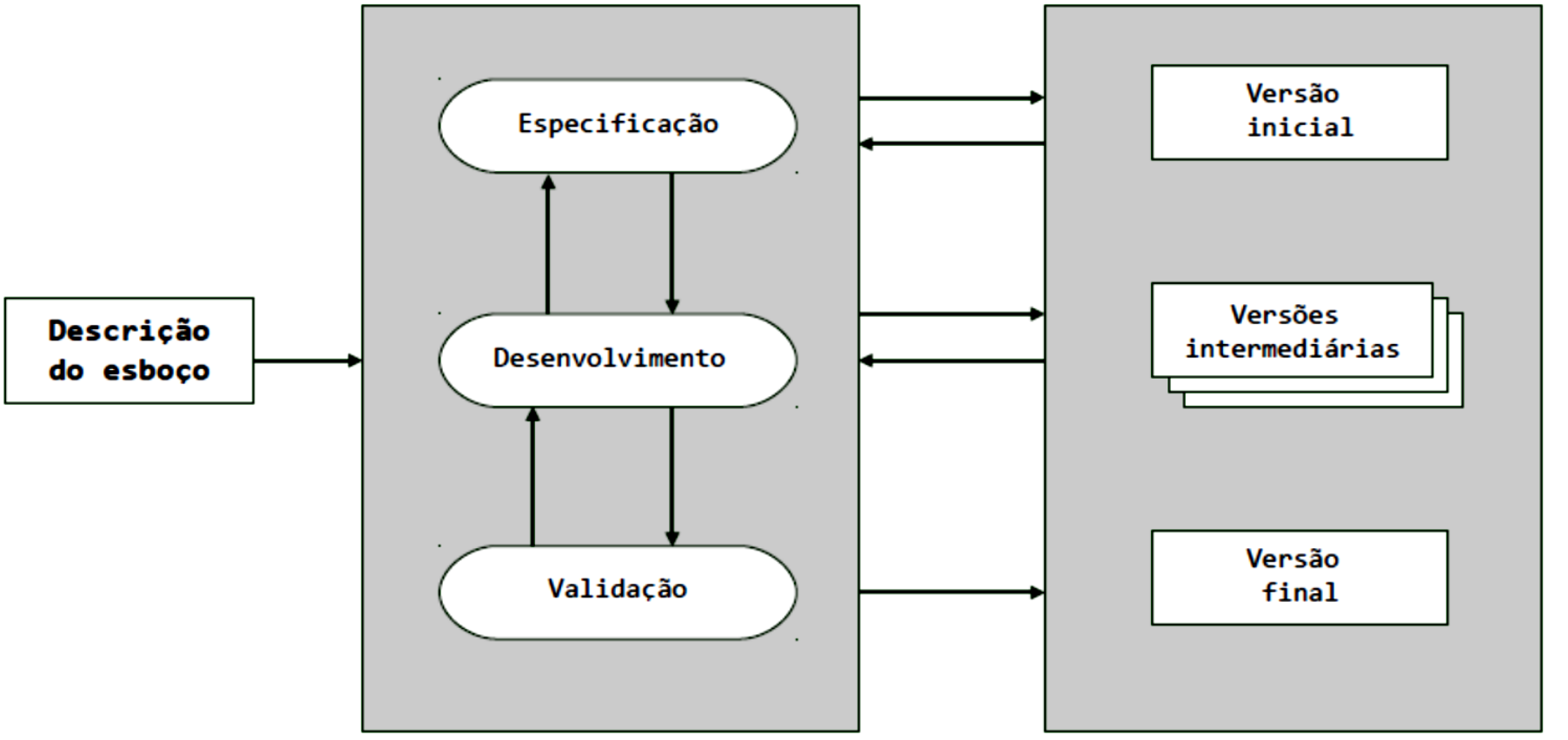
Modelo Cascata

- **Problemas**:
  - O resultado de cada fase envolve um ou mais documentos que são aprovados
    - **Gera muita documentação, nem sempre utilizada posteriormente**
  - A fase seguinte não deve iniciar ate que a fase precedente tenha sido concluída
  - **Particionamento inflexível** do projeto em estágios
    - Dificulta a resposta aos requisitos de mudança do cliente.
  - Apropriado somente quando os requisitos são bem compreendidos e quando as **mudanças são raras**
    - Poucos sistemas de negócio têm requisitos estáveis.

Modelo Evolucionário

- Implementação inicial, exposição do resultado aos comentários do usuário e refinamento desse resultado em versões
- As fases de Especificação, desenvolvimento e validação são **intercalas**
  - Um rápido *feedback* por meio dessas atividades

Modelo Evolucionário



Modelo Evolucionário

- **Vantagens**:
  - Especificação desenvolvida de forma incremental
  - Produz sistemas que atendam as necessidades imediatas dos clientes
- **Problemas**:
  - Processo não é visível
  - Sistemas podem ser mal estruturados devido à mudança contínua
  - Incorporar modificações torna-se cada vez mais difícil e oneroso

Prototipação

- Desenvolvimento rápido de um sistema
- 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;

Prototipação

Prototipação

- Vantagens:
  - Melhoria na facilidade de uso do sistema;
  - Maior aproximação do sistema com as necessidades dos usuários;
  - Melhoria da qualidade do projeto;
  - Melhoria na facilidade de manutenção;
  - Redução no esforço de desenvolvimento

Prototipação

- Problemas:
  - O protótipo pode não ser necessariamente usado da mesma forma que o sistema final;
  - O testador do protótipo pode não ser um usuário típico do sistema;

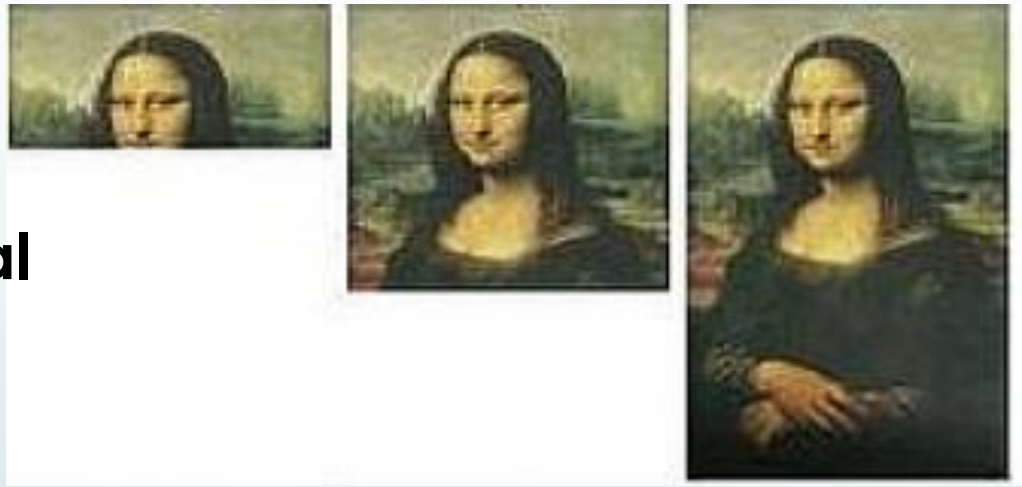
Processos Iterativos

- Necessidade de utilizar **diferentes abordagens para diferentes partes**, de maneira que um **modelo hibrido** tem de ser utilizado;
- Necessidade de **iteração**, em que partes do processo são repetidas, a medida que os requisitos do *software* evoluem;
- Pode ser aplicado a qualquer um dos **modelos genéricos de processo**;
- Dois modelos:
  - **Incremental**
  - **Espiral**

Processos Iterativos

Entrega 1 **Entrega 2** **Entrega 3**

Incremental

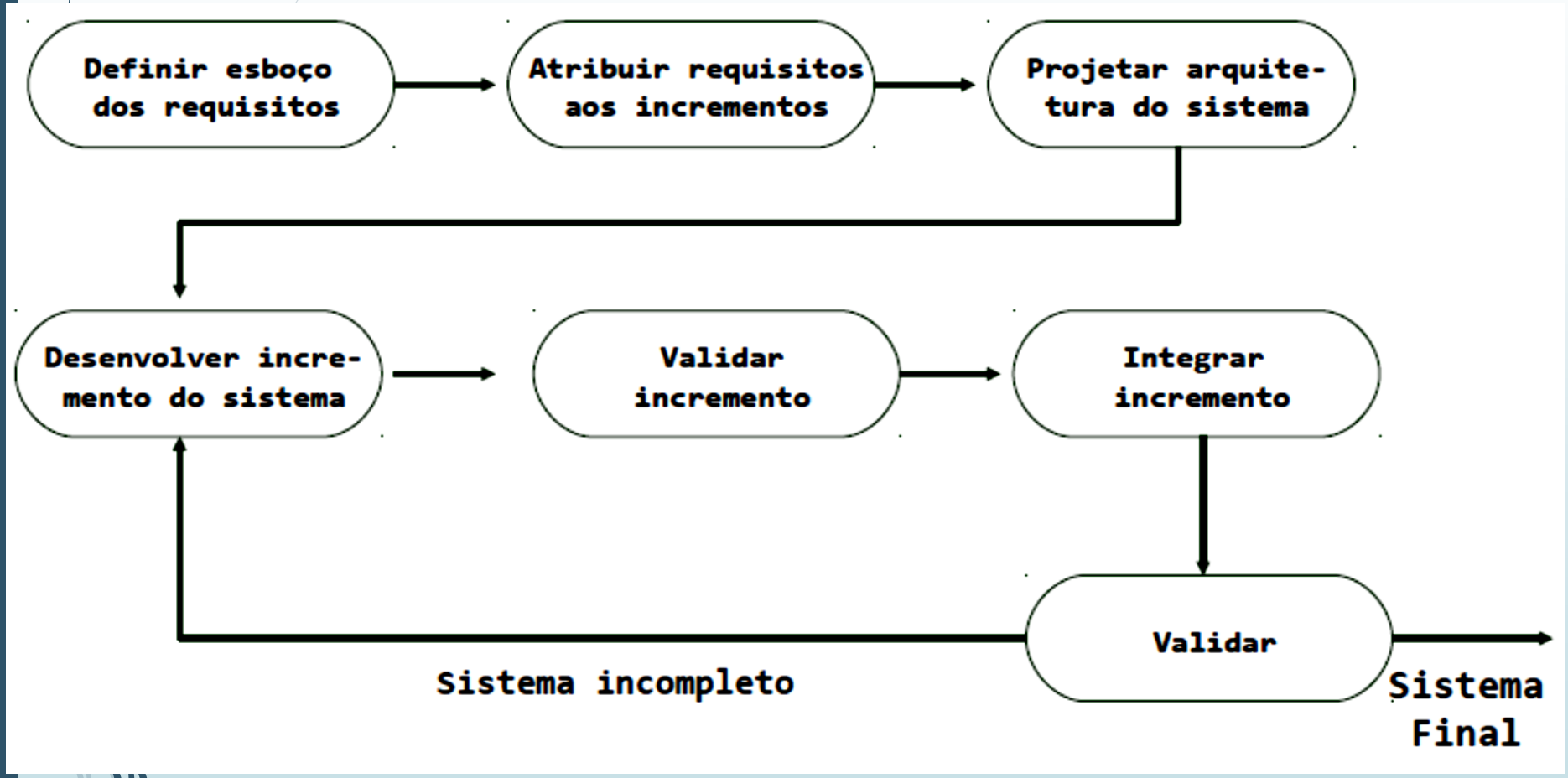


Espiral

Processos Iterativos – Incremental

- O sistema é entregue ao cliente em **incrementos**
  - Cada incremento fornece parte da funcionalidade
- Os requisitos são **priorizados**
  - Requisitos de prioridade mais alta são incluídos nos incrementos iniciais.
- Uma vez que o desenvolvimento de um incremento é iniciado, os requisitos são congelados
  - Os requisitos para os incrementos posteriores podem continuar evoluindo (e incluir requisitos já implementados!)
- Uma vez que um incremento é concluído e entregue, os usuários podem colocá-lo em operação

Processos Iterativos – Incremental



Processos Iterativos – Incremental

- **Vantagens:**
  - Incrementos podem ser entregues regularmente ao cliente e, desse modo, a funcionalidade de sistema é disponibilizada mais **cedo**;
  - Os incrementos iniciais agem como **protótipos** para elicitar os requisitos para incrementos posteriores do sistemas;
  - **Riscos menores** de falha geral do projeto;
  - Os serviços de sistema de mais **alta prioridade** tendem a receber **mais testes**

Processos Iterativos – Incremental

- **Problemas:**
  - Os incrementos devem ser relativamente pequenos, e cada incremento deve produzir alguma funcionalidade para o *software;*
  - Dificuldade de identificar os recursos comuns exigidos por todos os incrementos

Processos Iterativos – Espiral

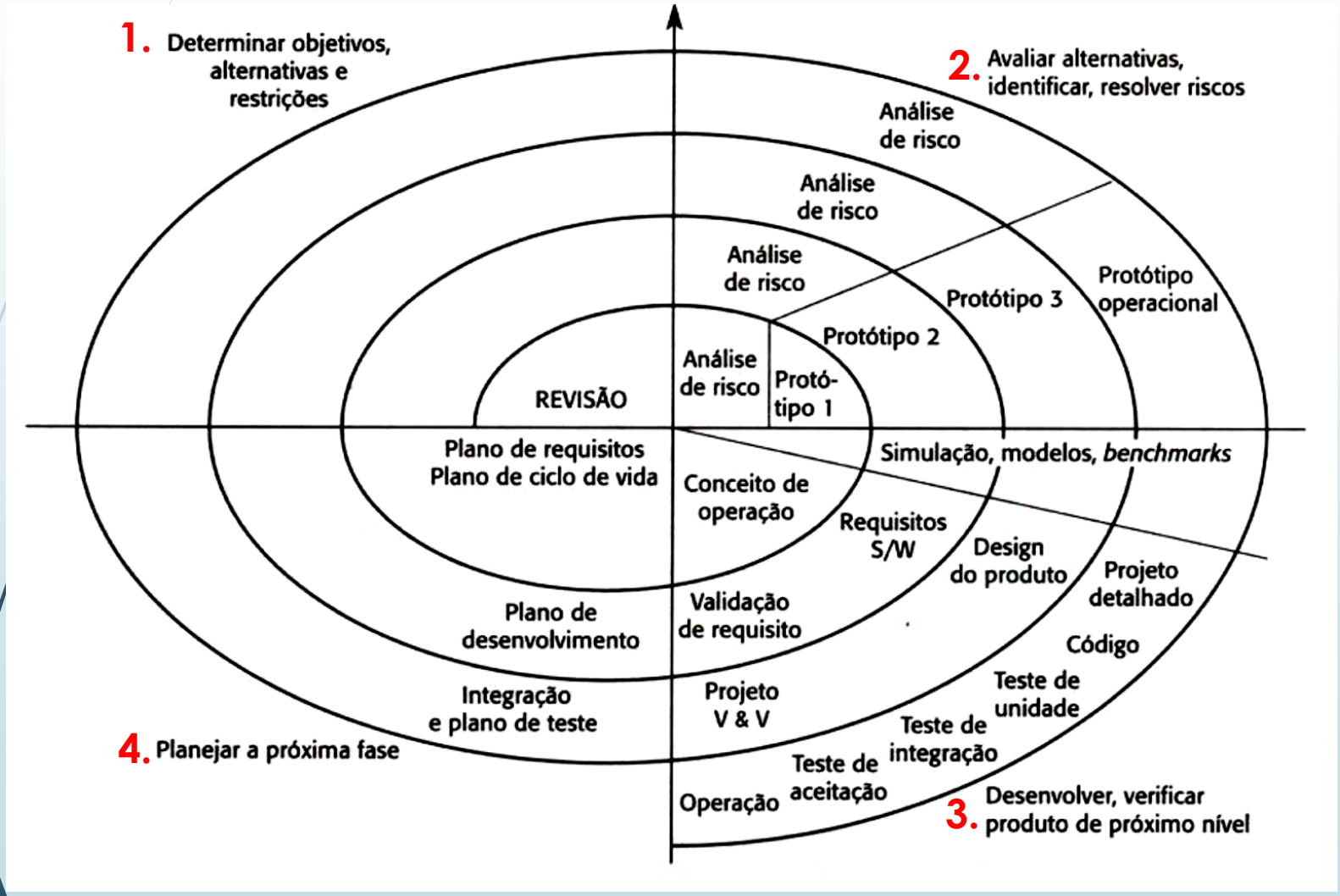
- Representa o processo de *software* como uma espiral
- Cada ***loop*** na espiral representa uma **fase do processo de *software***
  - Ex: o *loop* mais interno pode estar relacionado a **viabilidade do *software***
- Sem fases definidas, tais como especificação ou projeto – os *loops* na espiral são escolhidos dependendo do que é requisitado.
- Os **riscos** são explicitamente avaliados e resolvidos ao longo do processo

Processos Iterativos – Espiral

- Possui 4 setores:

1. Definição dos objetivos	• Objetivos específicos para a fase são identificados
2. Avaliação e redução de riscos	• Riscos são avaliados e atividades são realizadas para reduzir os riscos-chave
3. Desenvolvimento e validação	• Um modelo de desenvolvimento para o sistema, que pode ser qualquer um dos modelos genéricos, é escolhido
4. Planejamento	• O projeto é revisado e a próxima fase da espiral é planejada

Processos Iterativos – Espiral



Cascata x Evolucionário

CRITÉRIOS	CASCATA	EVOLUCIONÁRIO
Resistente à Mudanças	X	
Planejamento pré-determinado	X	X
Foco na documentação	X	X
Comunicação com o cliente	X	X
Prioridade das necessidades de acordo com o cliente		X
Testes unitários	X	X
Software testado apenas no final	X	

Ferramentas Case (*Computer – Aided SoftwareEngineering*)

- Ferramentas **automatizadas**;
- Auxiliam o desenvolvedor de sistemas em uma ou várias etapas do ciclo de desenvolvimento de software;
- Padronizam as modelagens desenvolvidas;
- Verificam a consistência, integridade e metodologia de diagramas;
- Reduzem ou eliminam inúmeros problemas durante ciclo de desenvolvimento.

Ferramentas Case (*Computer – Aided SoftwareEngineering*)

FUNCIONALIDADES	FERRAMENTAS
Controle de Versão	GitHub, Subversion
Gerência de Projetos	MSProject
Modelagem	**Star UML, Astah**
Prototipagem	NetBeans, Pencil
Programação	Eclipse, NetBeans
Teste	xUnit
Documentação	Office, OpenOffice

Ferramentas Case (*Computer – Aided SoftwareEngineering*)

- **Vantagens:**
  - Qualidade no produto final
  - Produtividade
  - Agilizar o tempo para tomada de decisão
  - Menor quantidade de códigos de programação
  - Melhoria e redução de custos na manutenção
  - Agilidade no retrabalho do software
  - Maior facilidade para desenvolvimento

Ferramentas Case (*Computer – Aided SoftwareEngineering*)

- **Desvantagens:**
  - Incompatibilidade de ferramentas
  - Conhecimento para utilização