Instituto Federal de Educação, Ciência e Tecnologia da Bahia - IFBA
Departamento de Ciência da Computação
Tecnólogo em Análise e Desenvolvimento de Sistemas

Modelos de processos de software

André L. R. Madureira <andre.madureira@ifba.edu.br>
Doutorando em Ciência da Computação (UFBA)
Mestre em Ciência da Computação (UFBA)
Engenheiro da Computação (UFBA)

Modelos de Processos de software

- São abstrações dos processos de software que explicam diferentes abordagens de desenvolvimento de sistemas
  - Não detalham atividades específicas
  - Podem ser ampliados e adaptados para criar processos de engenharia específicos

Classificação de Modelos de software

- **Dirigidos a planos**
  - Todas as atividades são planejadas com antecedência
  - O progresso é medido em relação ao planejamento inicial (cronograma de execução)
- **Processos ágeis (ou metodologia ágil)**
  - Planejamento gradativo
  - Ajustes nos processos são fáceis de serem realizados conforme demanda do cliente muda

Modelos de Processos de software

- Os modelos mais utilizados são:
  - **Modelo em cascata (ou baseado em ciclo de vida)**
  - **Desenvolvimento incremental**
  - **Engenharia de software orientada a reúso**

Modelos de Processos de software

- Os modelos mais utilizados são:
  - **Modelo em cascata (ou baseado em ciclo de vida)**
    - Atividades sequenciais (fases distintas)
  - **Desenvolvimento incremental**
    - Atividades intercaladas (incrementos de software)
  - **Engenharia de software orientada a reúso**
    - Reutilização de componentes de um sistema existente

Modelos de Processos de software

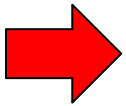
- Os modelos mais utilizados são:
  - **Modelo em cascata (ou baseado em ciclo de vida)**
  - **Desenvolvimento incremental**
  - **Engenharia de software orientada a reúso**

Esses modelos podem ser usados em conjunto
(não são mutuamente exclusivos)

Modelos de Processos de software

- Os modelos mais utilizados são:
  - **Modelo em cascata (ou baseado em ciclo de vida)**
  - **Desenvolvimento incremental**
  - **Engenharia de software orientada a reúso**

Esses modelos podem ser usados em conjunto
(não são mutuamente exclusivos)



Combinar as melhores características de cada modelo
(a depender da necessidade)

Exemplo de Uso Simultaneo de diferentes Modelos

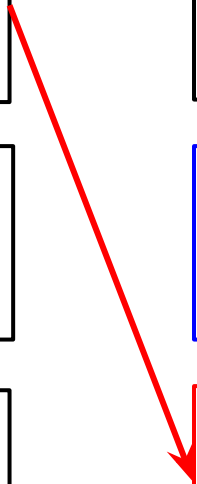
Ex: Subsistemas (módulo NFe, etc) (código semelhante para sistemas diferentes)	**Modelo em cascata**
Ex: Interface gráfica (GUI) (difícil de especificar com precisão)	**Desenvolvimento incremental**
Ex: Projeto da arquitetura do sistema (precisa da especificação completa)	**Engenharia de software orientada a reúso**

Exemplo de Uso Simultaneo de diferentes Modelos

Ex: Subsistemas (módulo NFe, etc) (código semelhante para sistemas diferentes)

Ex: Interface gráfica (GUI) (difícil de especificar com precisão)

Ex: Projeto da arquitetura do sistema (precisa da especificação completa)

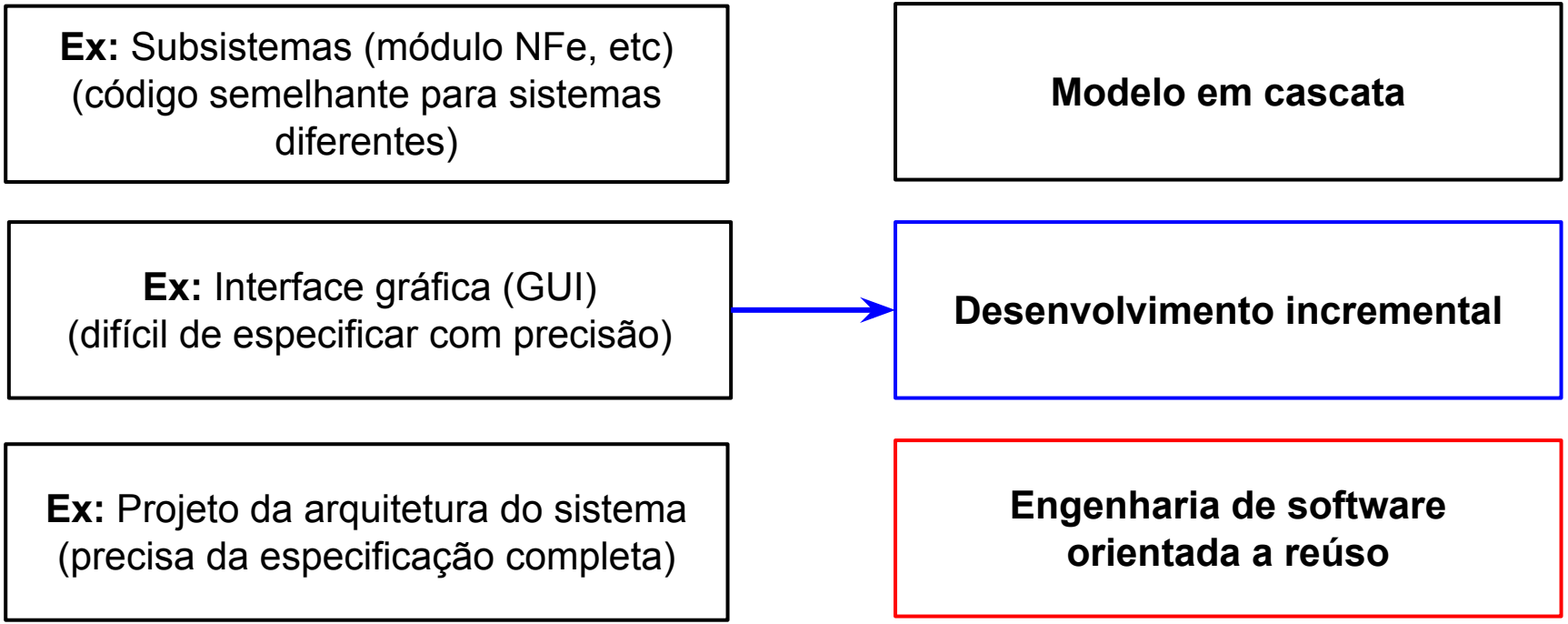


Modelo em cascata

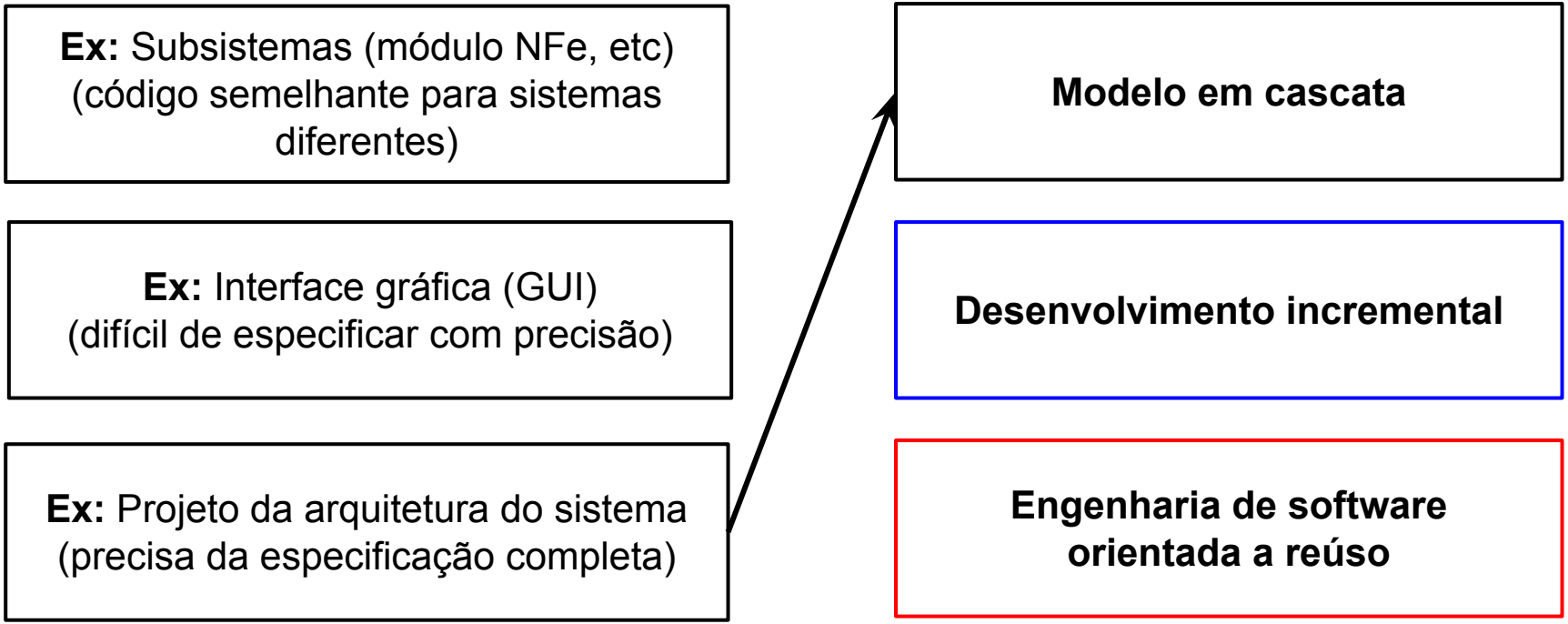
Desenvolvimento incremental

Engenharia de software orientada a reúso

Exemplo de Uso Simultaneo de diferentes Modelos



Exemplo de Uso Simultaneo de diferentes Modelos



Modelos em cascata (ou ciclo de vida de software)

- Também conhecido como **modelo *waterfall***
- Modelo **dirigido a planos**
- Atividades sequenciais, como fases distintas
  - **Resultado de cada fase**: um ou mais documentos aprovados
  - A fase seguinte não deve ser iniciada até que a fase anterior seja concluída

Modelos em cascata (ou ciclo de vida de software)

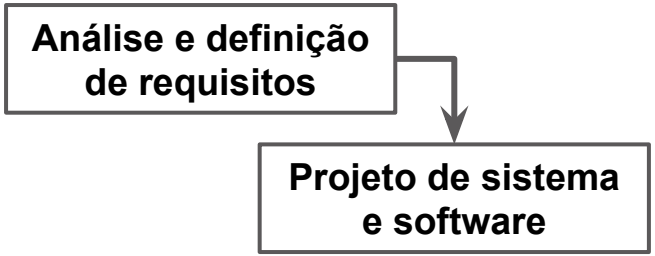
- Também conhecido como **modelo *waterfall***
- Modelo **dirigido a planos**
- Atividades sequenciais, como fases distintas
  - **Resultado de cada fase**: um ou mais documentos aprovados
  - A fase seguinte não deve ser iniciada até que a fase anterior seja concluída
  - Custo de elaboração de documentos para cada fase é alto
    - **Solução**: Congelar fases (não trabalhar mais nelas)

Modelos em cascata (ou ciclo de vida de software)

Análise e definição de requisitos

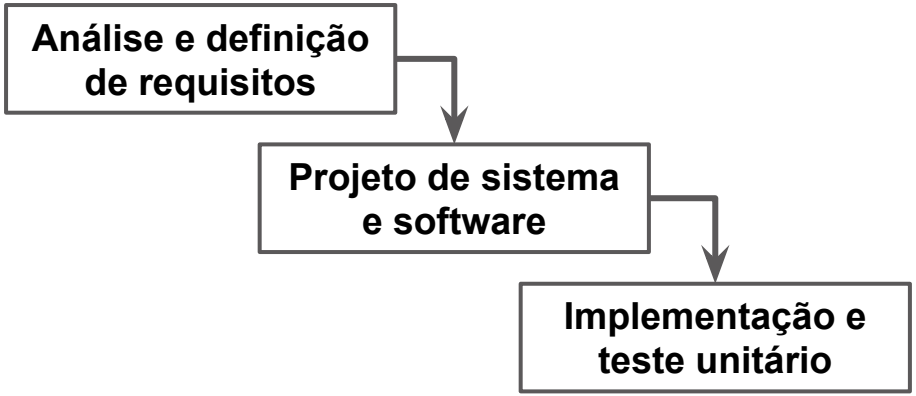
Consulta aos usuários para definir serviços, restrições e metas do sistema

Modelos em cascata (ou ciclo de vida de software)



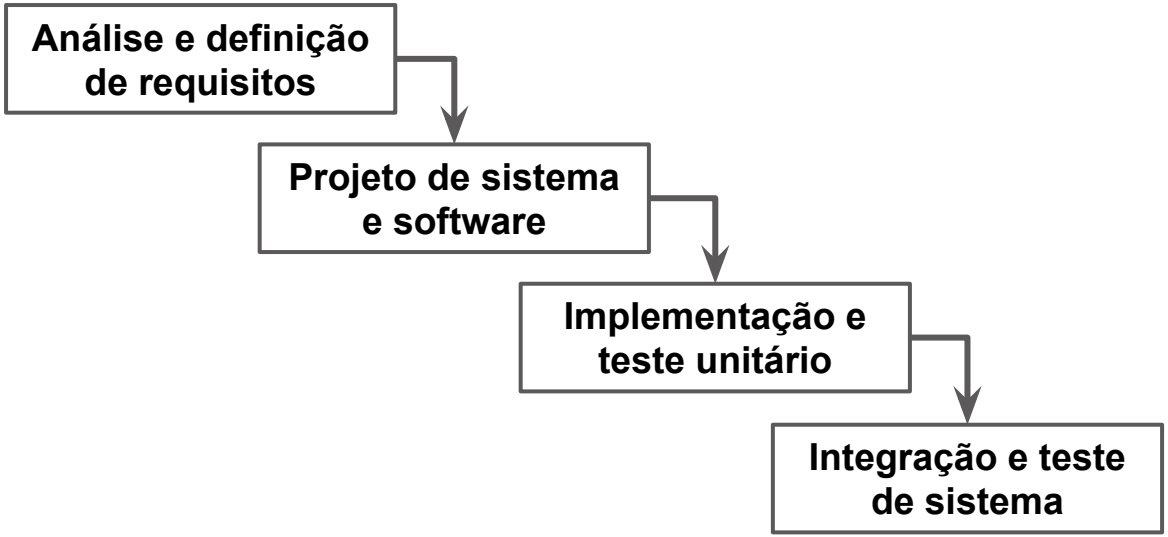
Definição da arquitetura geral do sistema

Modelos em cascata (ou ciclo de vida de software)



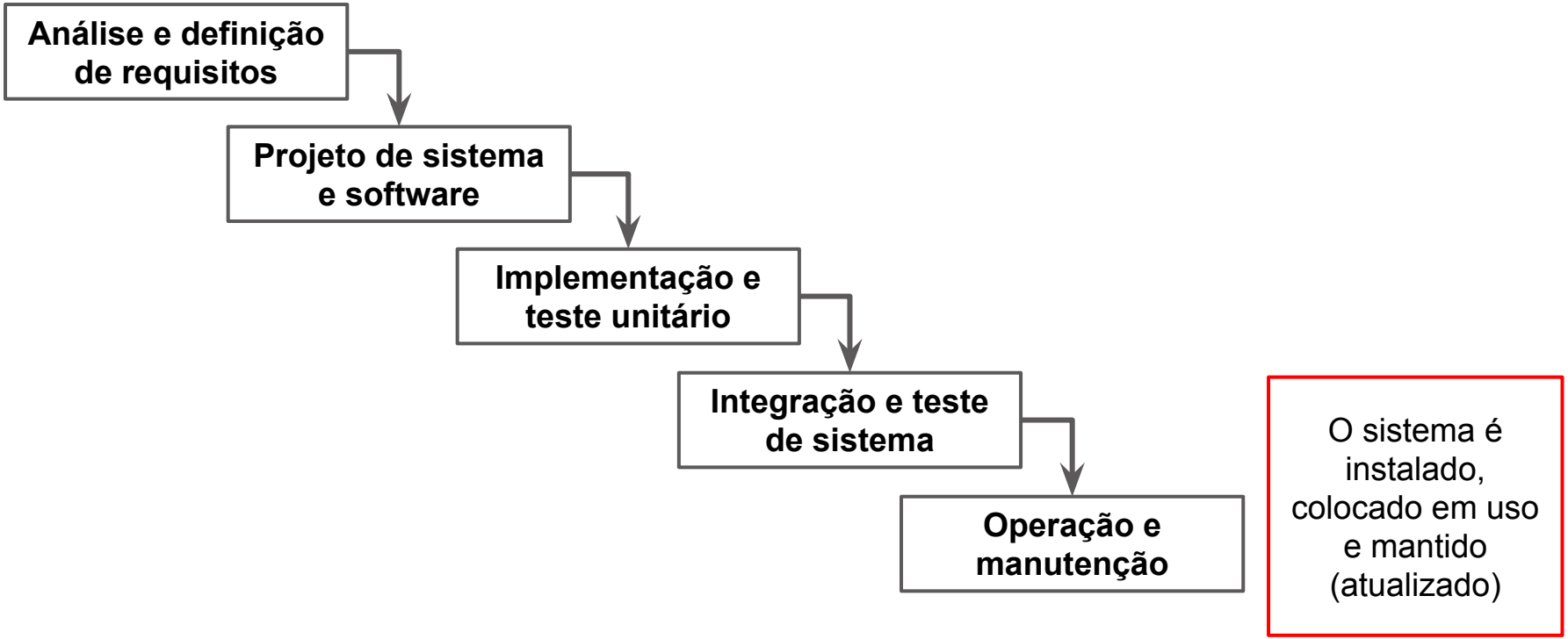
Desenvolvimento do projeto e teste de cada unidade que compõe o sistema

Modelos em cascata (ou ciclo de vida de software)

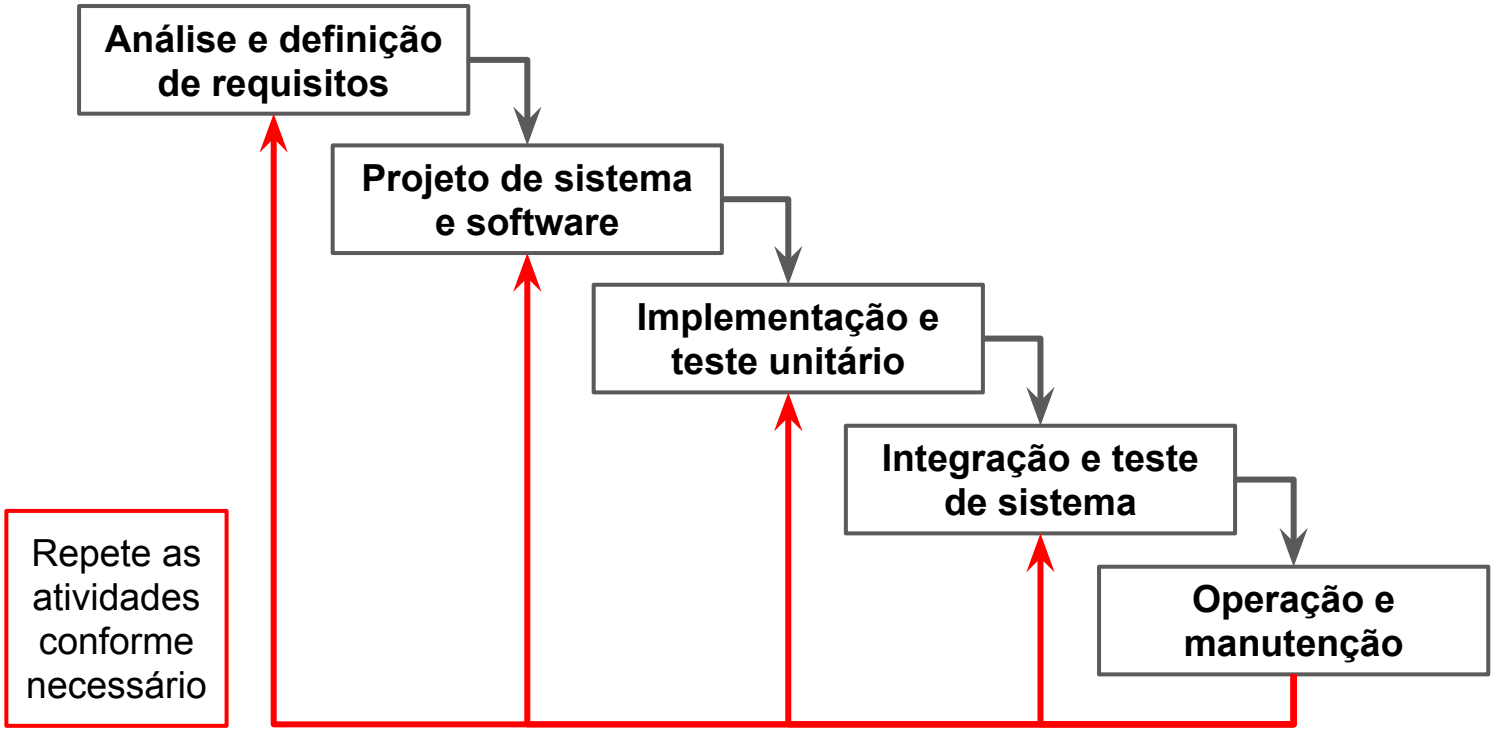


Integração das unidades para compor o sistema. **Testes com o sistema. Entrega para o usuário.**

Modelos em cascata (ou ciclo de vida de software)



Modelos em cascata (ou ciclo de vida de software)



Modelos em cascata (ou ciclo de vida de software)

- **Vantagens**:
  - Facilita o acompanhamento do progresso do projeto
  - Melhora a confiabilidade de sistemas complexos
    - **Ex**: sistemas embarcados de aeronaves

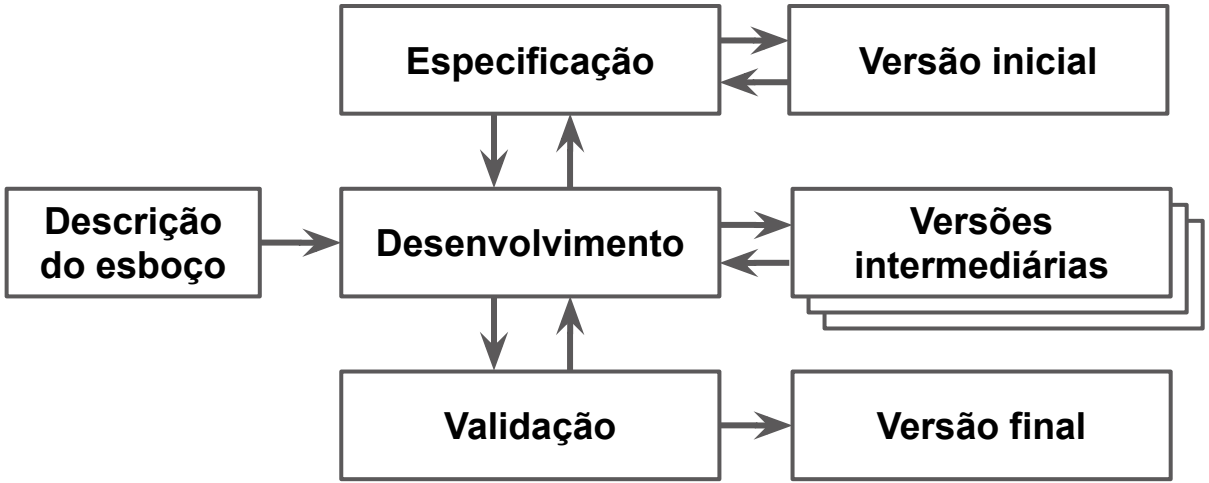
Modelos em cascata (ou ciclo de vida de software)

- **Vantagens**:
  - Facilita o acompanhamento do progresso do projeto
  - Melhora a confiabilidade de sistemas complexos, ou de vida-longa
    - **Ex**: sistemas embarcados de aeronaves
- **Desvantagens**:
  - Custo alto na elaboração de documentos e nos ajustes na especificação
  - Congelar fases mitiga os problemas acima, mas compromete a qualidade final do projeto

Desenvolvimento incremental

- Modelo baseado em **processos ágeis**
- Atividades intercaladas
- Sistema construido atraves de incrementos de software
  - Desenvolver uma implementação inicial
  - Criar várias versões, adicionando correções e funcionalidades
- Permite *feedback* rápido dos usuários em todas as atividades

Desenvolvimento incremental



Desenvolvimento incremental

- **Vantagens**:
  - Baixo custo na elaboração de documentos e nos ajustes na especificação
  - Facilidade em obter *feedback* dos clientes
  - Entrega e implementação rápida de funcionalidades
- **Desvantagens**:
  - Dificulta o acompanhamento do progresso do projeto
  - A estrutura do sistema tende a se degradar com a adição de novos incrementos
    - Dificuldade e custo alto na adição de incrementos

Engenharia de software orientada a reúso

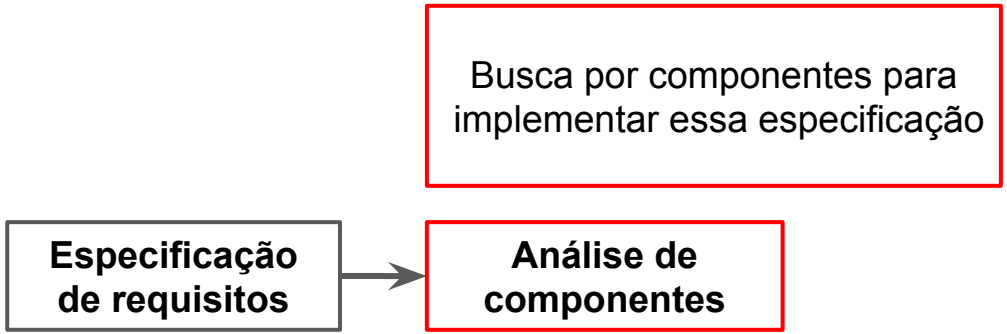
- Sistema construído atraves da integração e adaptação de componentes concebidos originalmente para outros sistemas
- Permite construção rápida de sistemas

Engenharia de software orientada a reúso

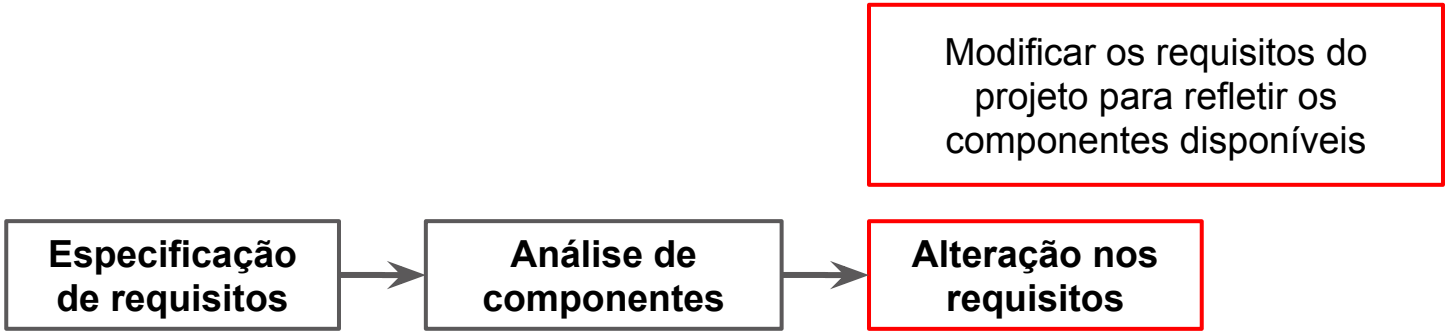
Consulta aos usuários para definir serviços, restrições e metas do sistema

Especificação de requisitos

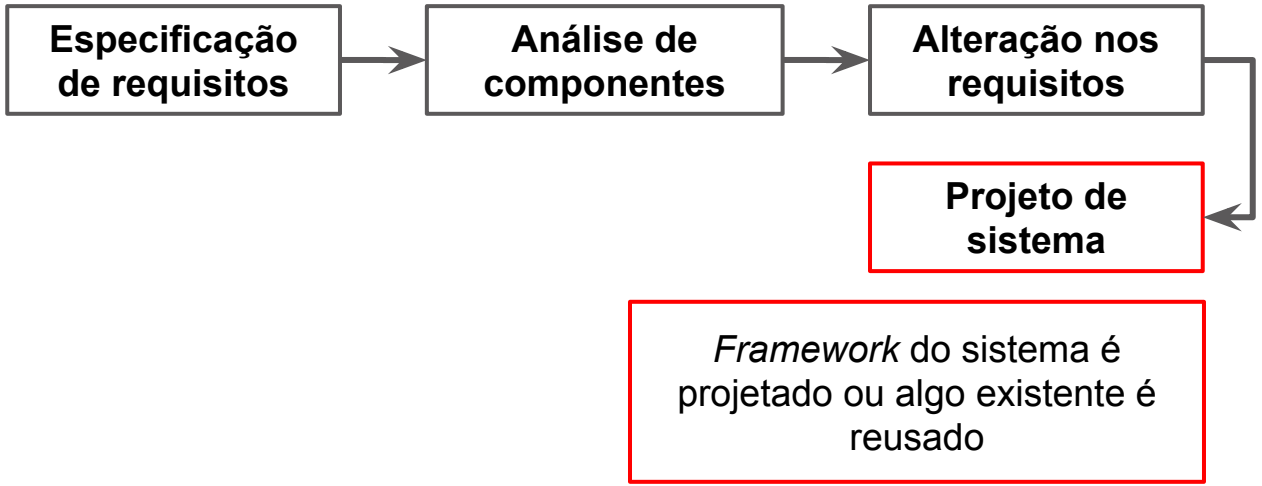
Engenharia de software orientada a reúso



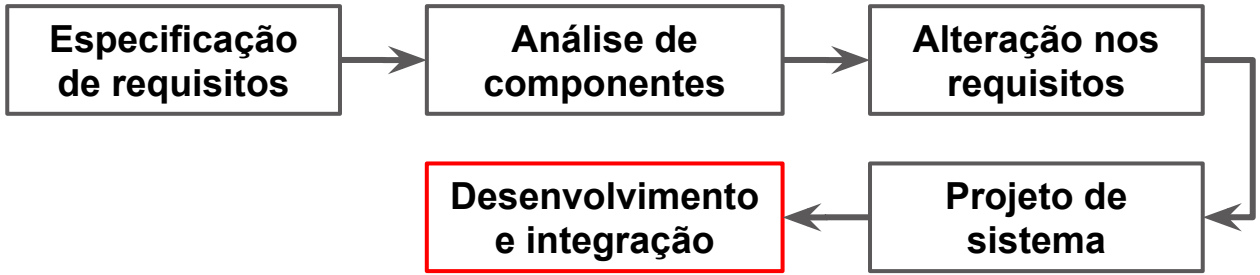
Engenharia de software orientada a reúso



Engenharia de software orientada a reúso

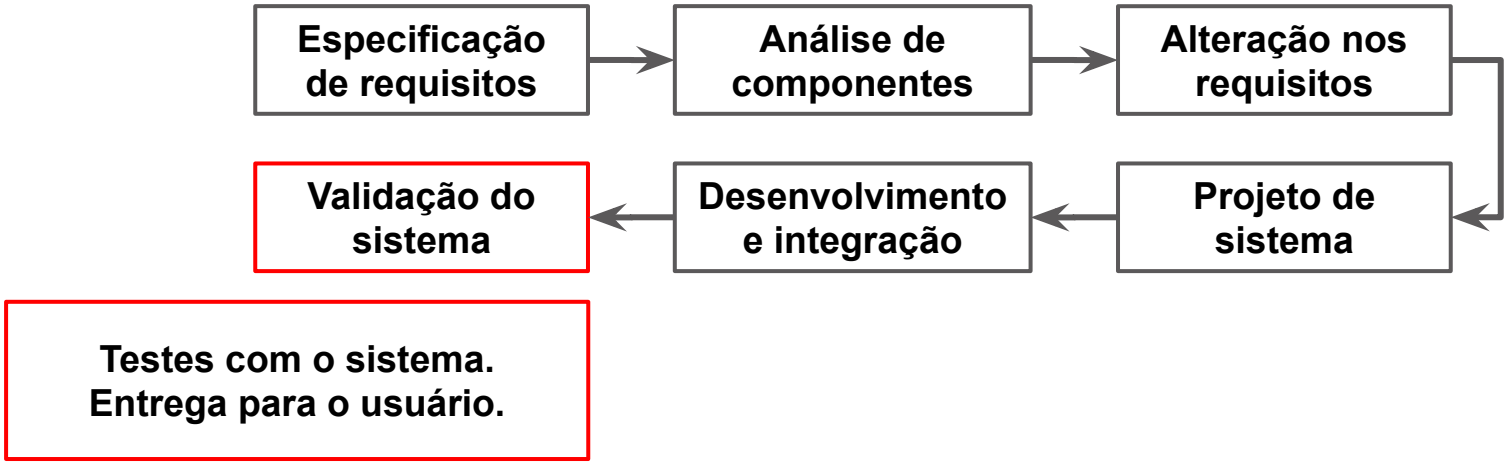


Engenharia de software orientada a reúso



Desenvolvimento dos softwares
que puderam ser reusados ou que não há compatibilidade com o projeto.
Integração dos componentes e sistemas.

Engenharia de software orientada a reúso



Engenharia de software orientada a reúso

- **Vantagens**:
  - Redução na quantidade de software a ser desenvolvido
    - Reduzir custos e riscos
  - Entrega mais rápida do software
- **Desvantagens**:
  - Exige modificação dos requisitos originais do sistema para permitir reuso de componentes
    - Sistema PODE não atender às necessidades dos usuários
  - Perda de controle sobre a evolução do sistema (código dos componentes)

Rational Unified Process (RUP)

- Modelo de processo moderno, derivado de trabalhos sobre a UML e o *Unified Software Development Process* (RUMBAUGH, et al., 1999; ARLOW e NEUSTADT, 2005)
  - Reúne elementos de todos os modelos de processo genéricos
  - Descreve as boas práticas de especificação e de projeto
  - Apoia a prototipação e a entrega incremental

Rational Unified Process (RUP)

- Atividades ou fases do modelo:

Concepção - *Rational Unified Process* (RUP)

Identificar as entidades externas (pessoas e sistemas) que vão interagir com o sistema e **definir** as interações

Concepção - *Rational Unified Process* (RUP)

Identificar as entidades externas (pessoas e sistemas) que vão interagir com o sistema e **definir** as interações

↓

Avaliar a contribuição do sistema para o negócio

Concepção - *Rational Unified Process* (RUP)

Identificar as entidades externas (pessoas e sistemas) que vão interagir com o sistema e **definir** as interações

Avaliar a contribuição do sistema para o negócio

Cancelar projeto
(baixa contribuição para o negócio)

Prosseguir com projeto
(contribuição significativa para o negócio)

Elaboração - *Rational Unified Process* (RUP)

Objetivos:
- Obter **compreensão do problema** dominante
- Estabelecer **framework da arquitetura** do sistema
- Desenvolver o **plano do projeto**
- Identificar os maiores **riscos do projeto**

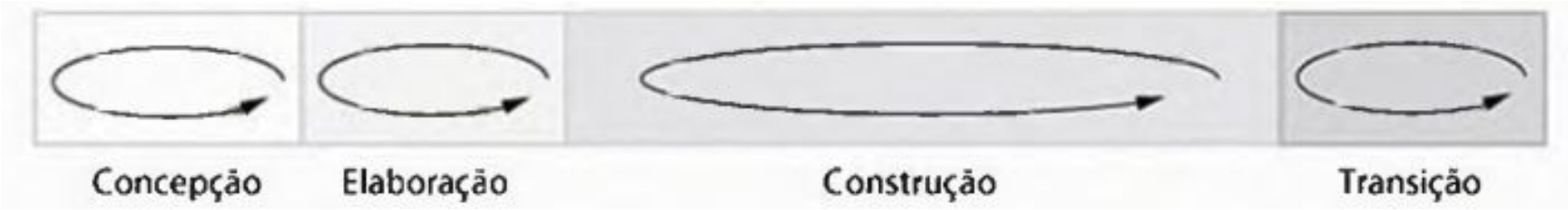
Produto (resultado): Modelo de requisitos
- **Ex:**
  - Diagrama de casos de uso da UML
  - Descrição da arquitetura
  - Plano de desenvolvimento do software

Construção - *Rational Unified Process* (RUP)

Envolve projeto, programação e testes do sistema, nos quais partes do sistema são desenvolvidas em paralelo e integradas

Produto (resultado): Sistema funcional e documentação

Transição - *Rational Unified Process* (RUP)



Implantação (deploy) do sistema no ambiente real dos usuários

Referencial Bibliográfico

- SOMMERVILLE, Ian. **Engenharia de Software**. 6. ed. São Paulo: Addison-Wesley, 2003.

- PRESSMAN, Roger S. **Engenharia de Software**. São Paulo: Makron Books, 1995.

- JUNIOR, H. E. **Engenharia de Software na Prática**. Novatec, 2010.