



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

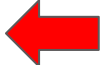
Processos de software

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)

Processos de software

- Conjunto de atividades relacionadas que levam à produção de um produto de software
 - **Especificação**
 - **Desenvolvimento**
 - Projeto e Implementação
 - **Verificação e Validação**
 - **Evolução**

Processos de software

- Conjunto de atividades relacionadas que levam à produção de um produto de software
 - **Especificação** 
 - **Desenvolvimento**
 - Projeto e Implementação
 - **Verificação e Validação**
 - **Evolução**

Especificação de software

- Envolve a compreensão, definição e identificação de:
 - Serviços requisitados do sistema
 - Restrições relativas à operação e ao desenvolvimento
- **Objetivo:** construir **especificação funcional** (documento de requisitos do sistema)
 - Mas o que seriam esses “requisitos”?

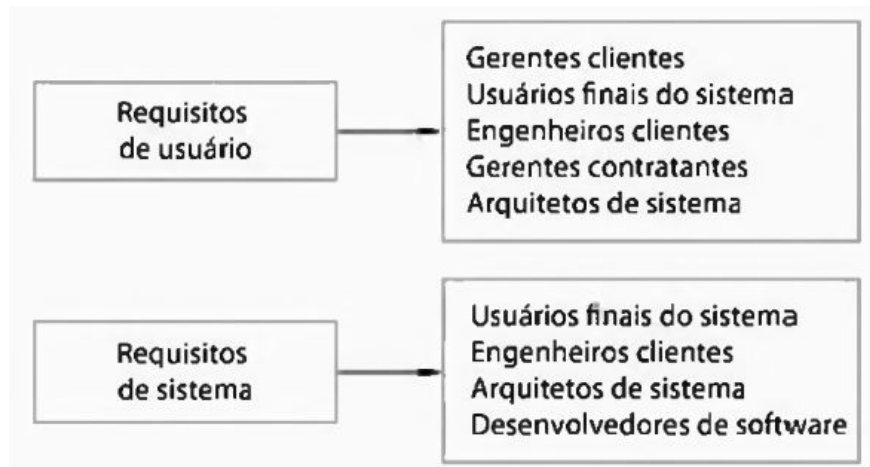
Requisitos de software

- Descrições do que o sistema deve fazer, os serviços que ele oferece e as restrições a seu funcionamento
- Os requisitos refletem as necessidades dos clientes
- Podem ser classificados de acordo com seu **grau de detalhamento**:
 - **Requisitos de usuário**: descrição abstrata de alto nível, usando linguagem natural com diagramas
 - **Requisitos do sistema**: descrição detalhada dos serviços e restrições do sistema, definindo exatamente o que deve ser implementado.

Requisitos de software

- Porque ter requisitos com diferentes níveis de detalhamento?
 - Diferentes pessoas têm diferentes necessidades de compreensão sobre um sistema

Ex: um gerente ou usuário de um sistema bancário estão mais preocupados com os serviços fornecidos pelo sistema, do que em como um sistema vai ser implementado



Exemplo de Requisitos de software

Requisitos de usuário:

1. O sistema deve gerar relatórios gerenciais mensais que mostrem o custo dos medicamentos prescritos por cada clínica durante aquele mês.

Requisitos de sistema:

- 1.1 No último dia útil de cada mês deve ser gerado um resumo dos medicamentos prescritos, seus custos e as prescrições de cada clínica.
- 1.2 Após 17:30h do último dia útil do mês, o sistema deve gerar automaticamente o relatório para impressão.
- 1.3 Um relatório será criado para cada clínica, listando os nomes dos medicamentos, o número total de prescrições, o número de doses prescritas e o custo total dos medicamentos prescritos.

Classificação de Requisitos de software

Requisitos funcionais (RF):

Descreve o comportamento do sistema perante determinadas entradas, bem como os serviços que ele deve fornecer

Requisitos não-funcionais (RNF):

São restrições impostas sobre os serviços e funções oferecidos pelo sistema.

Classificação de Requisitos de software

Requisitos funcionais (RF):

Descreve o comportamento do sistema perante determinadas entradas, bem como os serviços que ele deve fornecer

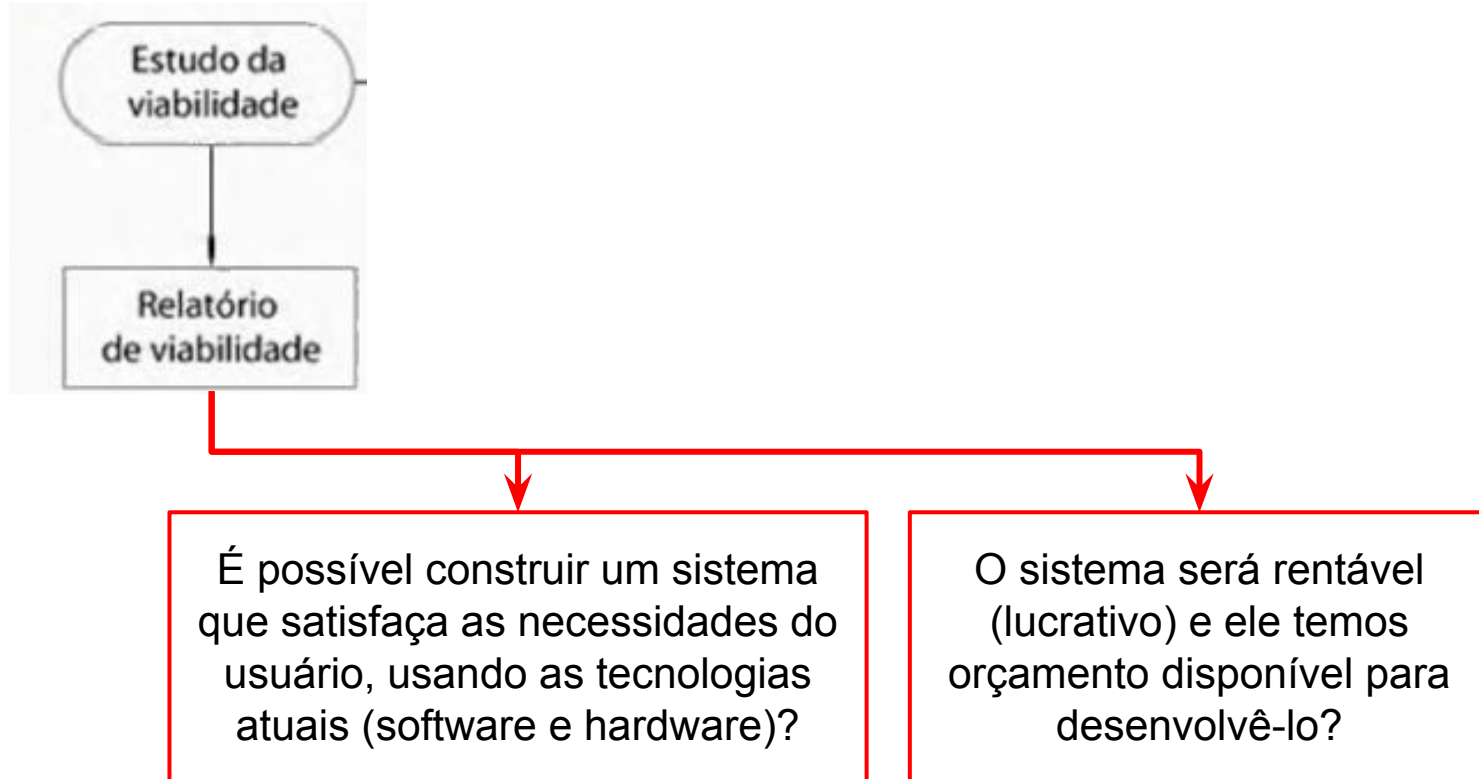
- **RF1:** Sacar dinheiro no caixa
 - **RF2:** Emitir extrato
 - **RF3:** Alterar senha
- **RF4:** Solicitar empréstimo
 - **RF5:** Investir dinheiro

Requisitos não-funcionais (RNF):

São restrições impostas sobre os serviços e funções oferecidos pelo sistema.

- **RNF1:** Sacar da conta-corrente só se saldo > 0
- **RNF2:** Alterar senha só se nova senha possuir 6 dígitos
- **RNF3:** Tempo máximo para ficar com App do banco aberto

Atividades da Especificação de software

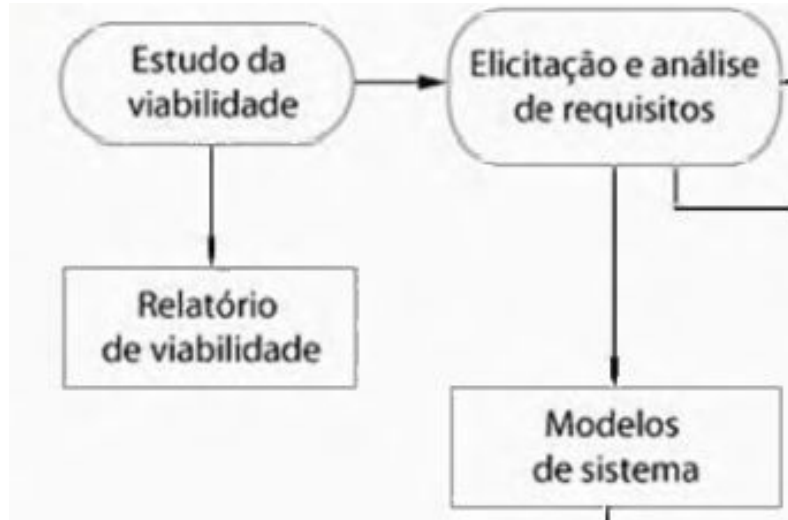


Atividades da Especificação de software



Esse estudo deve ser barato e rápido de ser executado
(tempo e dinheiro são recursos escassos)

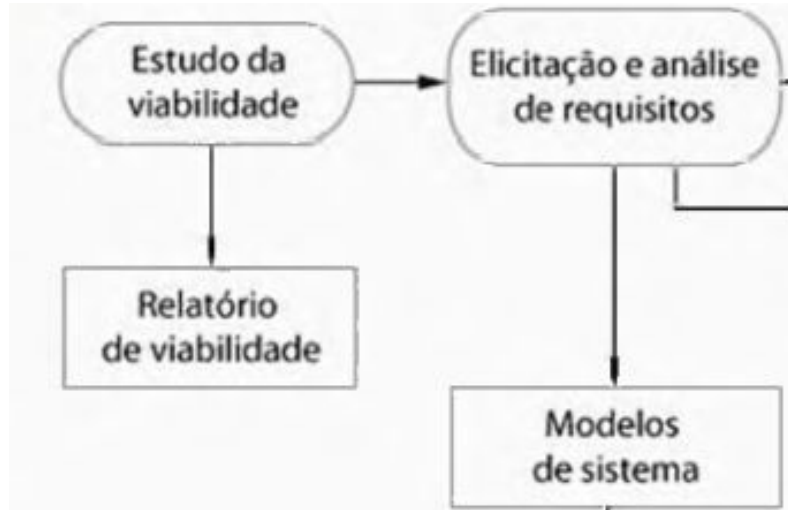
Atividades da Especificação de software



Derivar os requisitos do sistema por meio de:

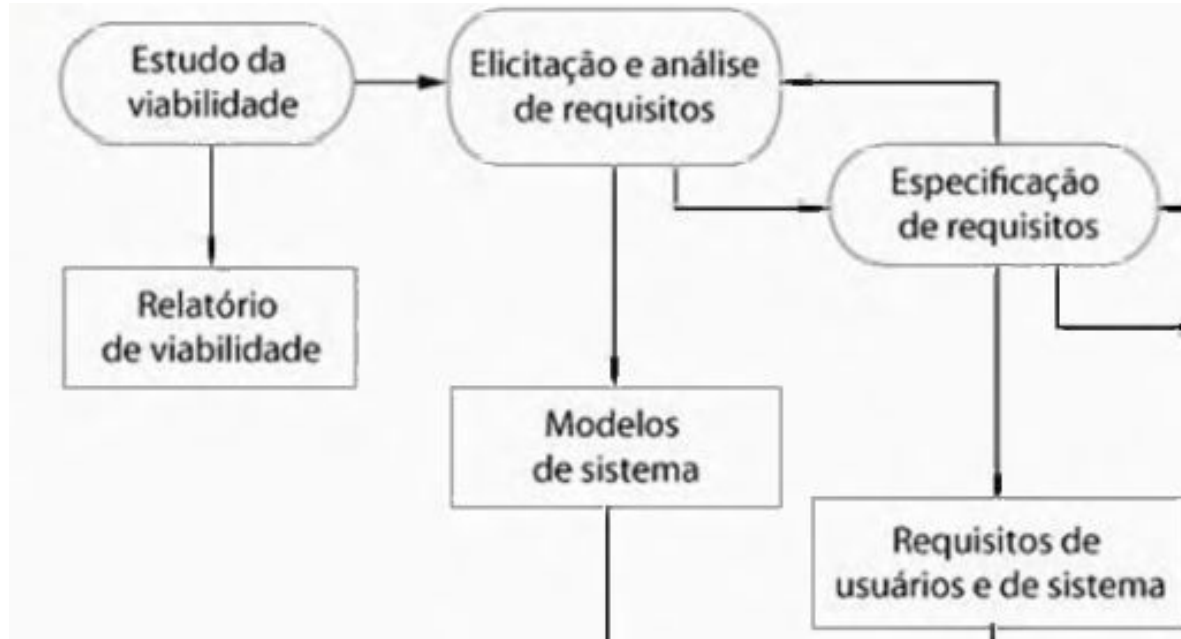
- Observação dos sistemas existentes
- Discussões com os potenciais usuários e compradores
- Análise de tarefas
- Demais etapas

Atividades da Especificação de software



Ajudam a entender o sistema proposto

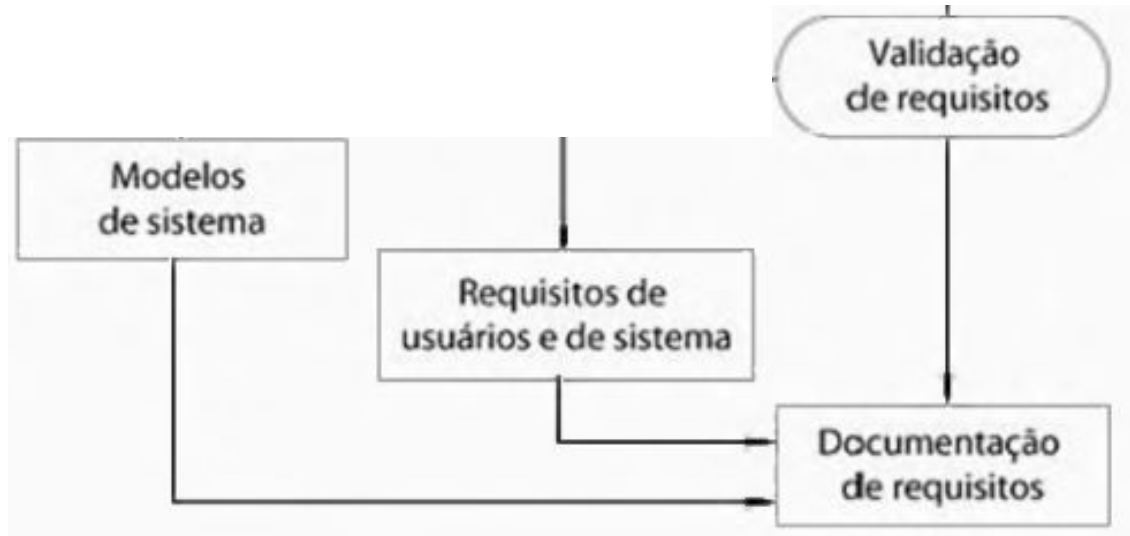
Atividades da Especificação de software



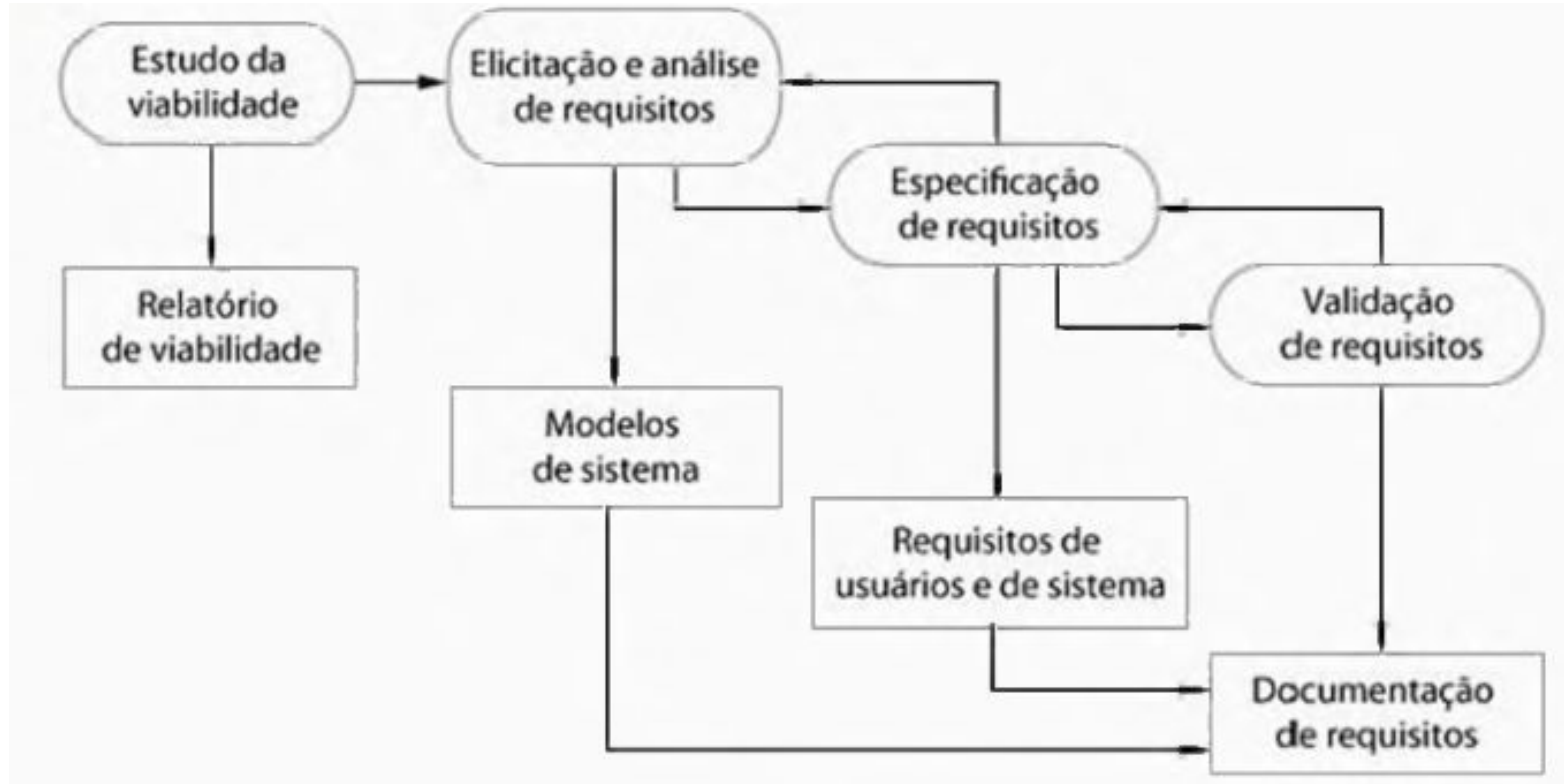
Traduzir as informações obtidas na atividade anterior em um conjunto de requisitos

Atividades da Especificação de software

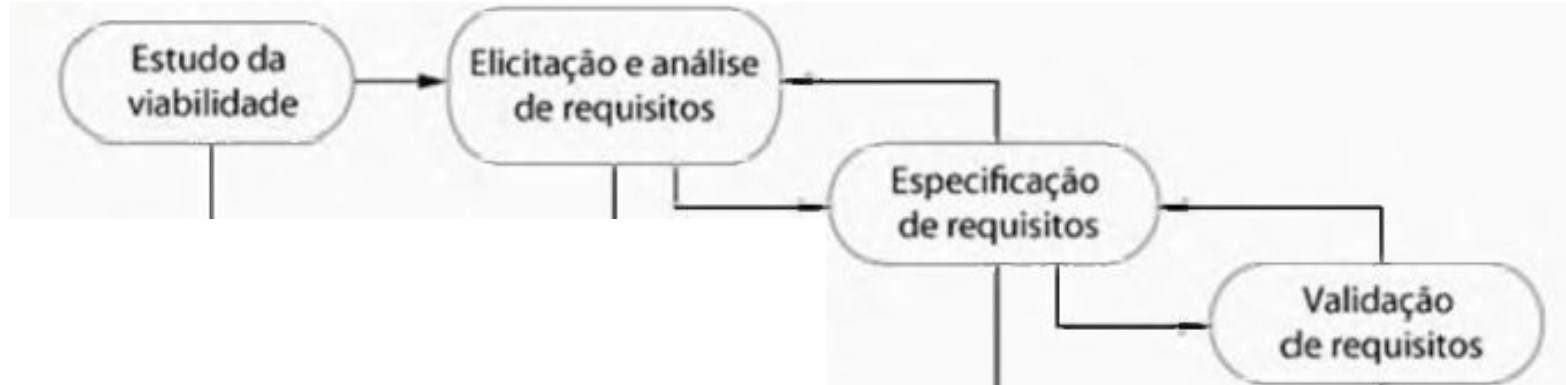
Verifica os requisitos quanto a **realismo**,
consistência e **completude**



Atividades da Especificação de software

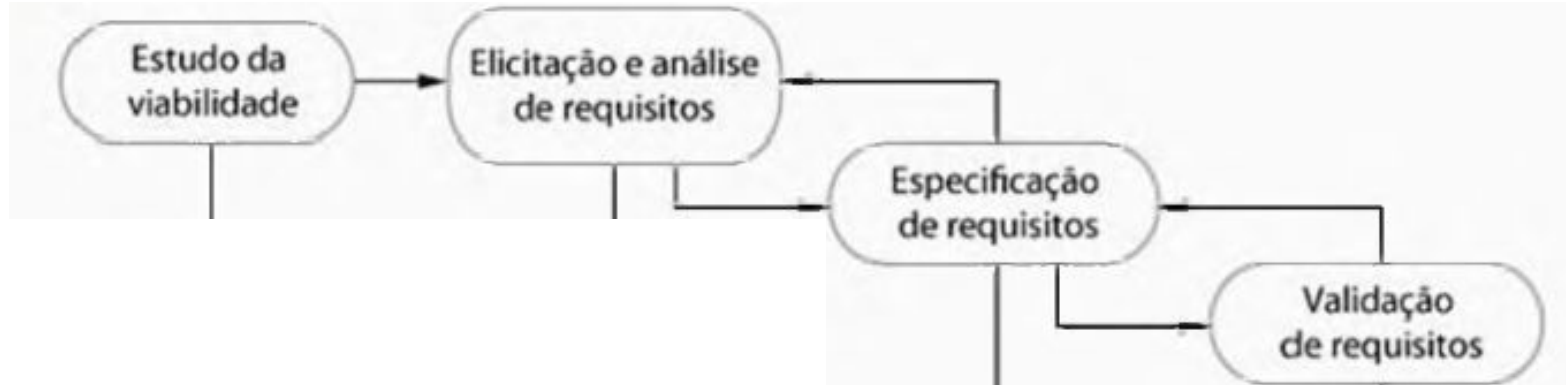


Atividades da Especificação de software



As atividades de **elicitação**, **especificação** e **validação** são re-executadas conforme novos requisitos emergem durante o processo.


Atividades da Especificação de software



As atividades de **elicitação**, **especificação** e **validação** são re-executadas conforme novos requisitos emergem durante o processo.

As atividades não são necessariamente sequenciais
(elas podem ser intercaladas)

Processos de software

- Especificação
- **Desenvolvimento** (Projeto e Implementação) 
- Verificação e Validação
- Evolução

Projeto e implementação de software

- Conversão da especificação em um sistema executável
- Composto por
 - **Projeto de software**
 - “Modelagem do sistema”
 - **Implementação do software**
 - “Codificação do sistema”

Projeto de software

- É a descrição da:
 - Estrutura do software
 - Modelos e estruturas de dados utilizados
 - Interfaces entre os componentes do sistema
 - Algoritmos usados

Projeto de software

- É a descrição da:
 - Estrutura do software
 - Modelos e estruturas de dados utilizados
 - Interfaces entre os componentes do sistema
 - Algoritmos usados

Processo iterativo

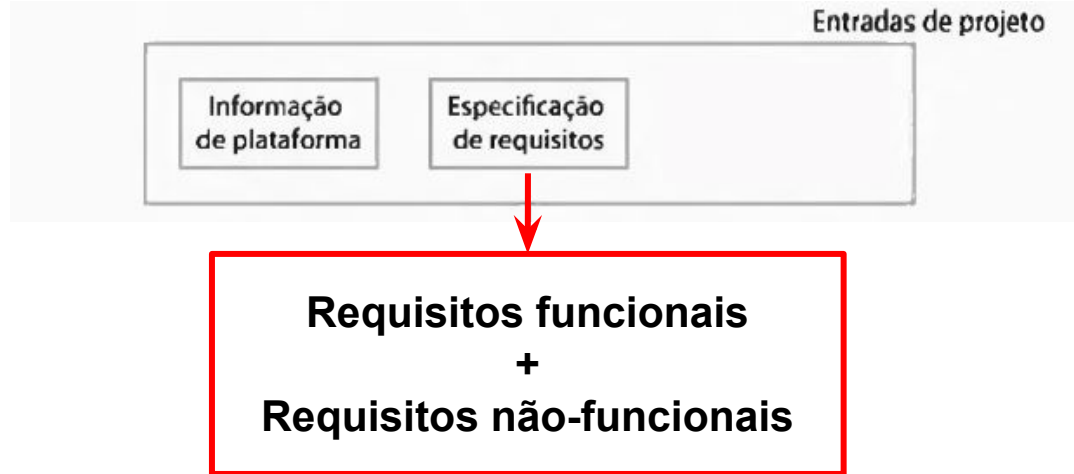
(adição de formalidade e detalhes, revisões constantes para correção de projetos anteriores)

Projeto de software

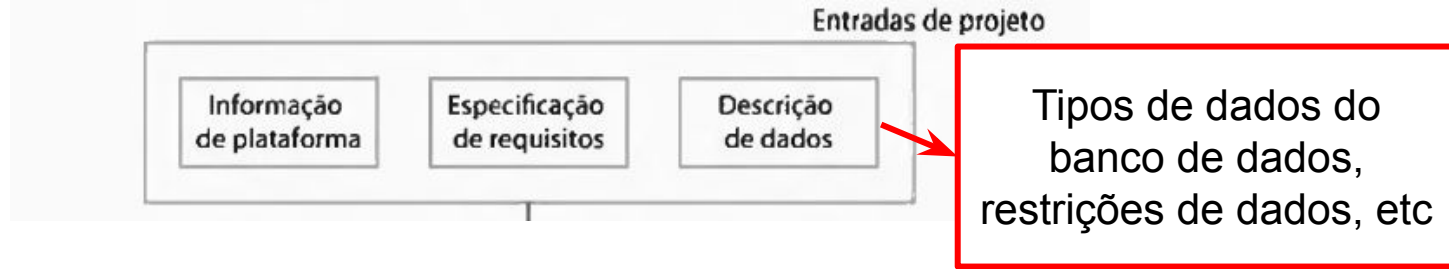
**Windows,
Linux,
MacOS,
...**



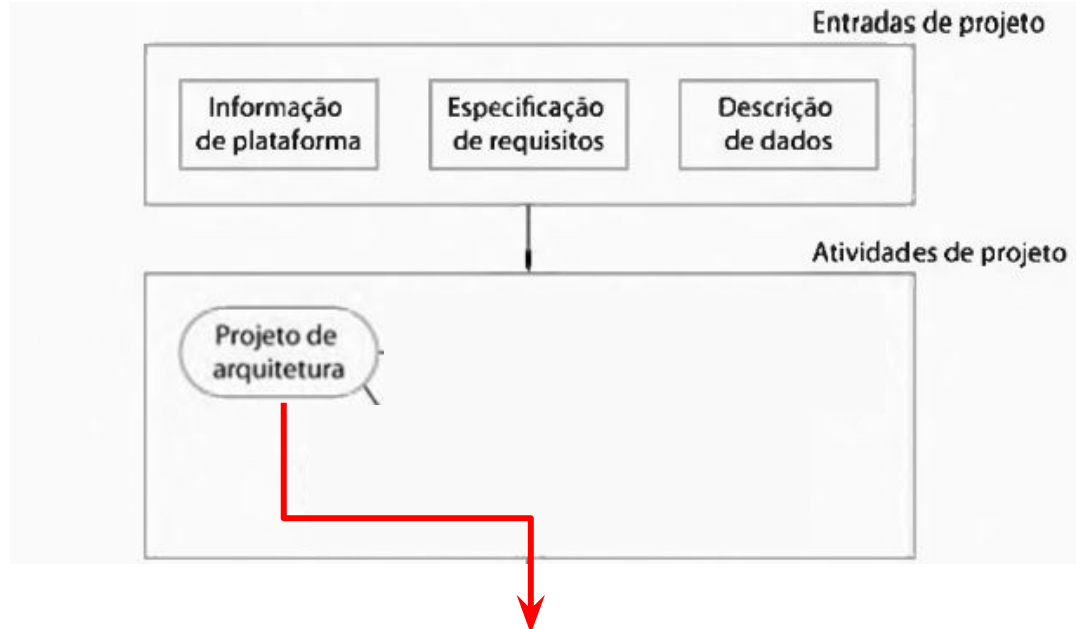
Projeto de software



Projeto de software

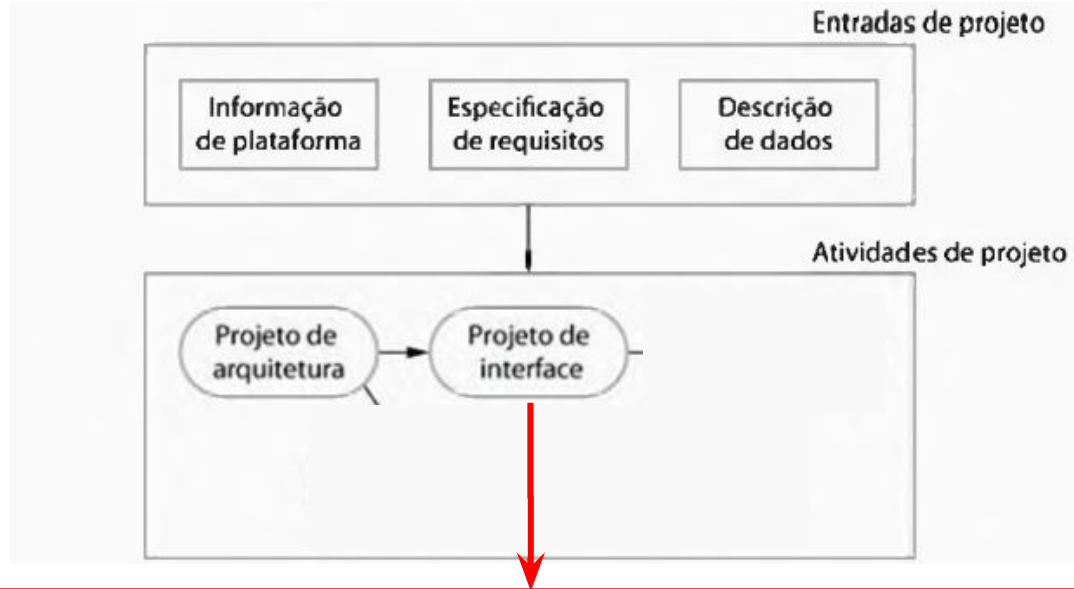


Projeto de software



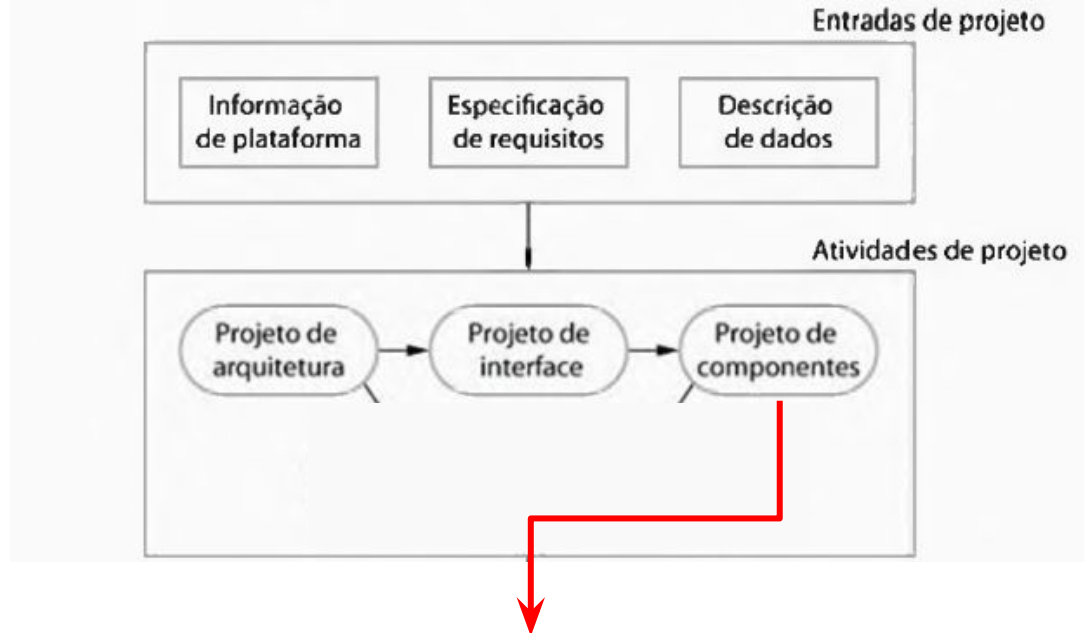
Identifica a estrutura geral do sistema, os componentes principais (subsistemas ou módulos), seus relacionamentos e como eles são distribuídos

Projeto de software



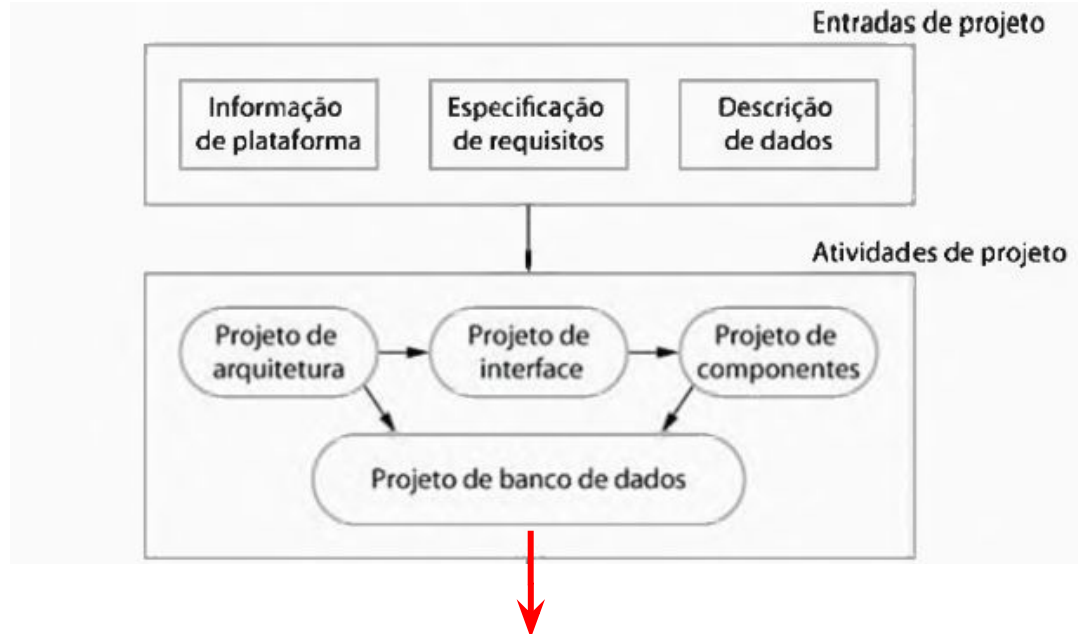
Define as interfaces entre os componentes do sistema, de maneira precisa e inequívoca, de forma a permitir o uso dos componentes entre si, sem que eles precisem conhecer como cada componente foi implementado

Projeto de software



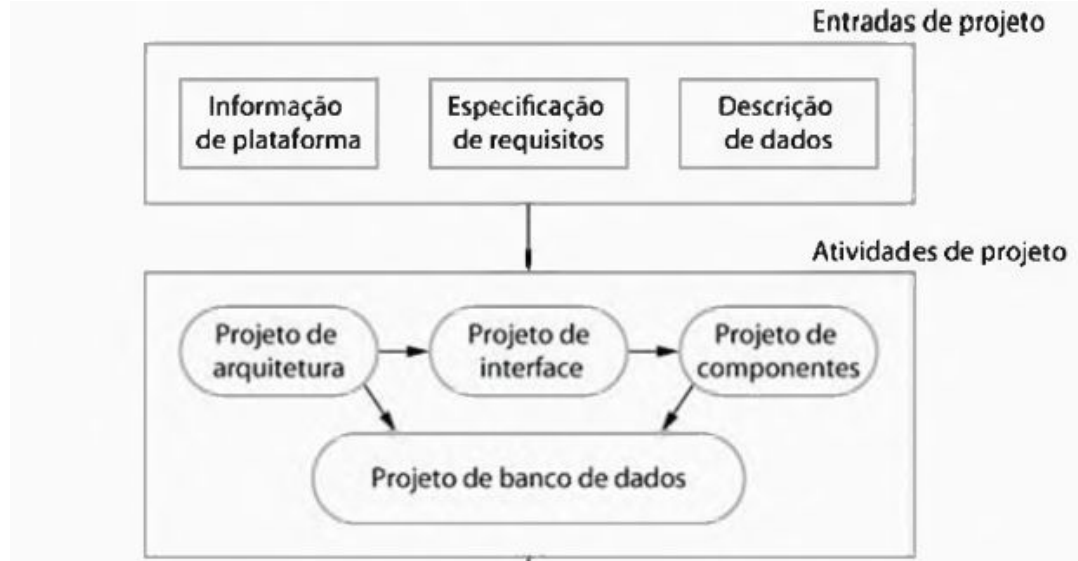
Define o **funcionamento de cada componente** do sistema, ou **listas de alterações** a serem feitas em cada componente reusável.

Projeto de software



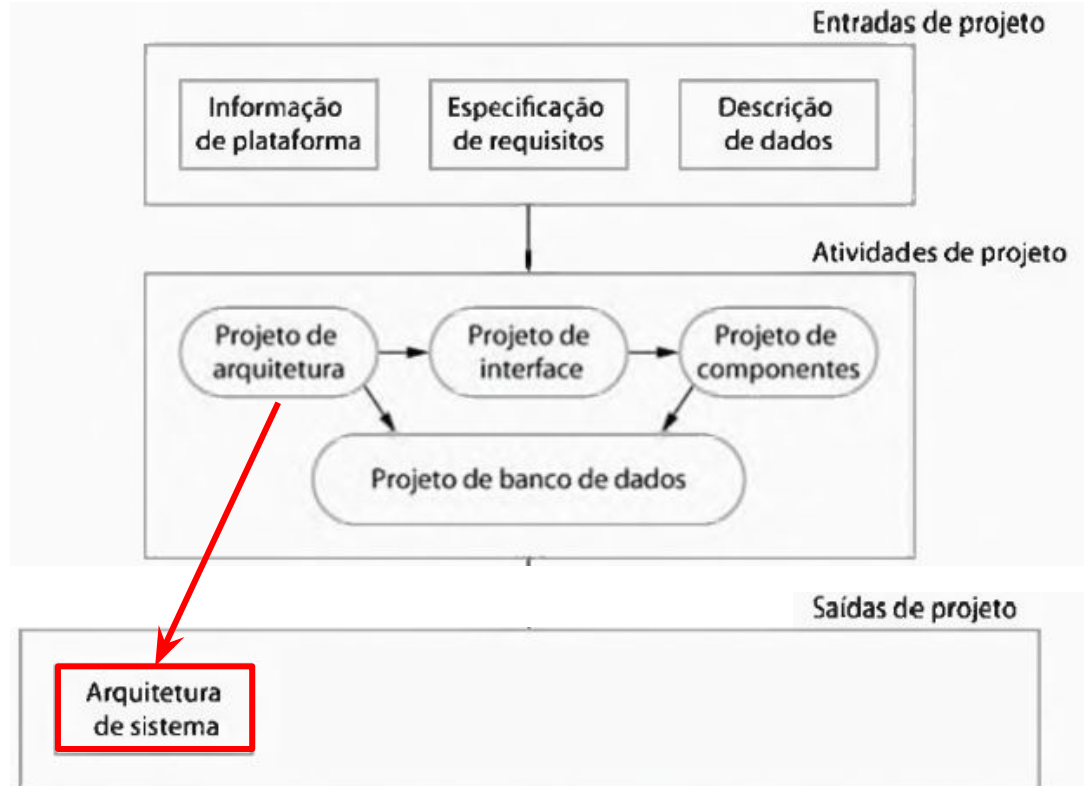
Define as estruturas de dados do sistema e a sua representação em um banco de dados.

Projeto de software

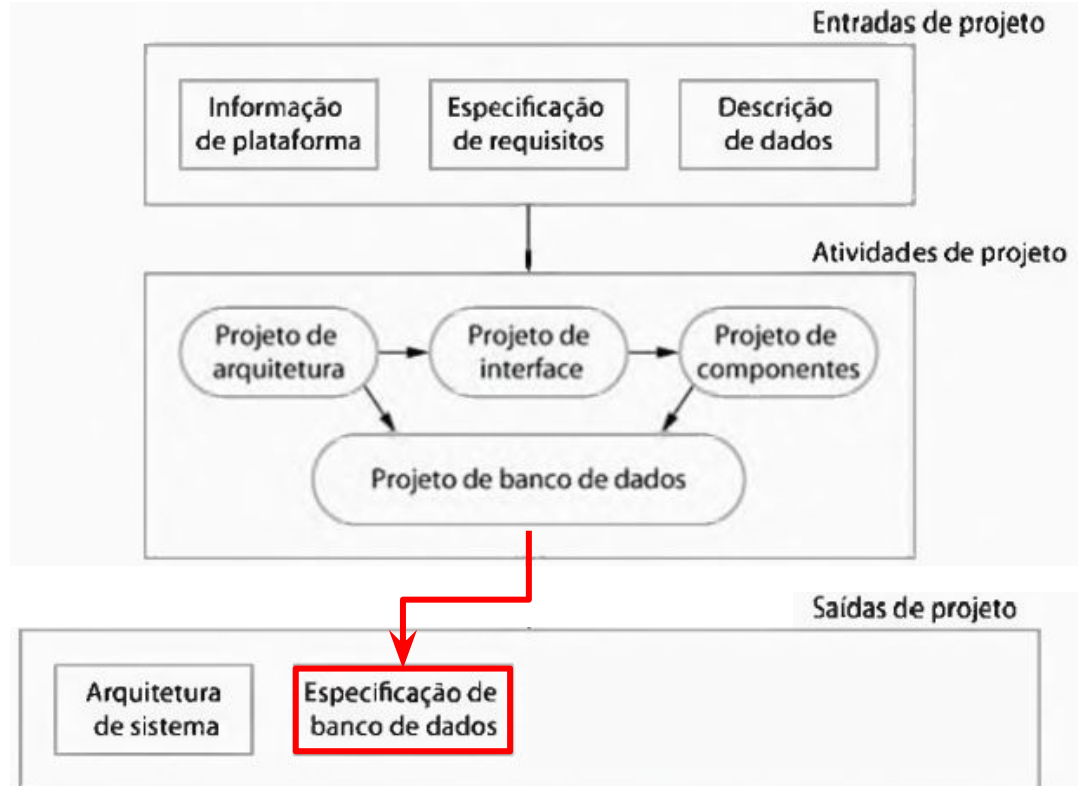


Apesar da representação sequencial acima,
as atividades de projeto podem ser INTERCALADAS

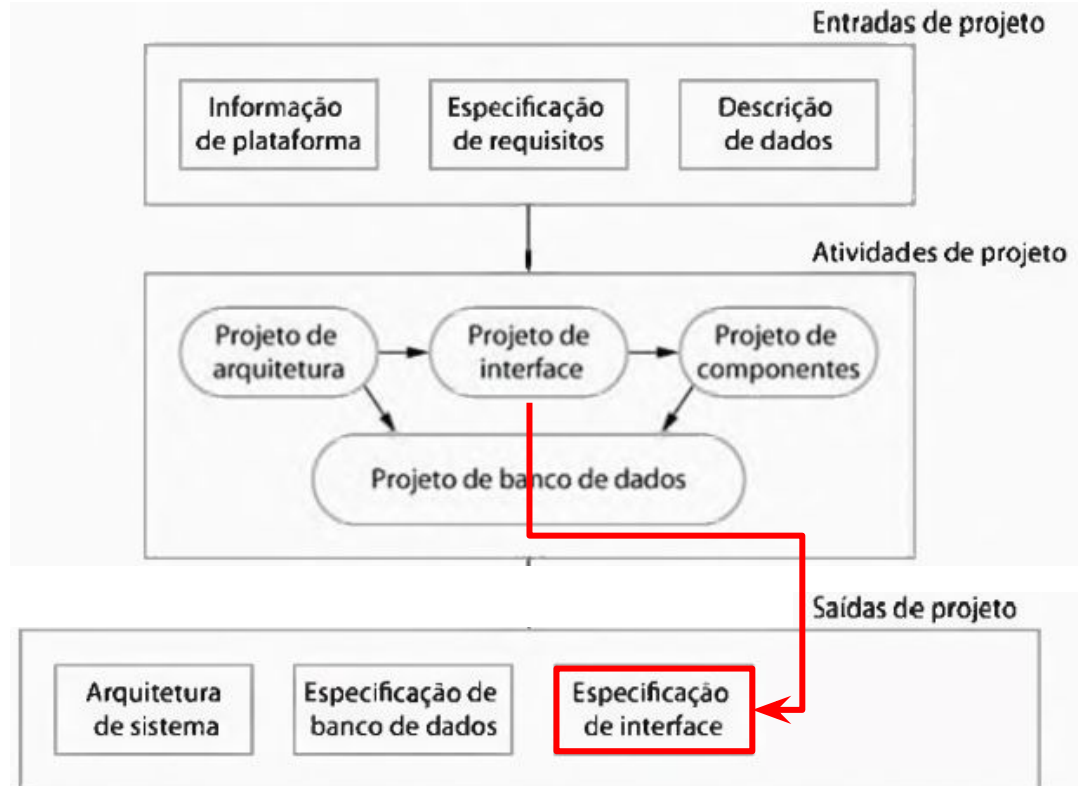
Projeto de software



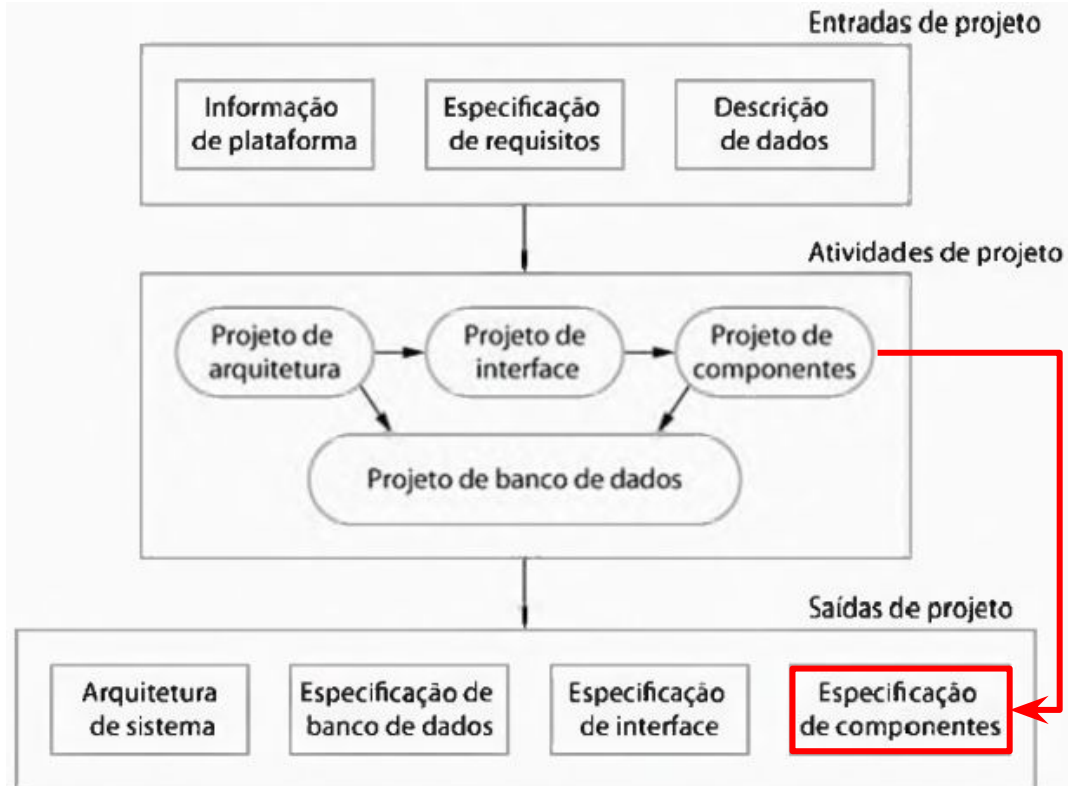
Projeto de software



Projeto de software



Projeto de software




Implementação de software

- Não existe um processo geral a ser seguido
 - Alguns programadores começam o desenvolvimento pelos componentes que eles compreendem (“mais fáceis”)
 - Outros fazem o oposto, desenvolvendo primeiro os componentes mais difíceis

Implementação de software

- Para identificar defeitos em um sistema, é necessário realizar **testes de defeitos**
- Os defeitos devem ser corrigidos através do ***debugging*** do código
 - ***Debugging***: Localização e correção de defeitos no código
 - Gerar hipóteses sobre o comportamento do programa
 - Testar essas hipóteses, na esperança de encontrar um defeito que tenha causado a saída anormal

Processos de software

- Especificação
- Desenvolvimento (Projeto e Implementação)
- **Verificação e Validação** 
- Evolução

Verificação e Validação de software

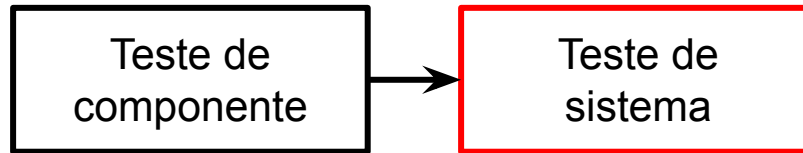
- **Objetivo:** Mostrar que um software:
 - Está adequado às suas especificações
 - Satisfaz as especificações do cliente do sistema
- **Como?**
 - Processos iterativos que envolvem
 - Testes de programa
 - Processos de verificação (revisões e inspeções)

Verificação e Validação de software

Teste de
componente

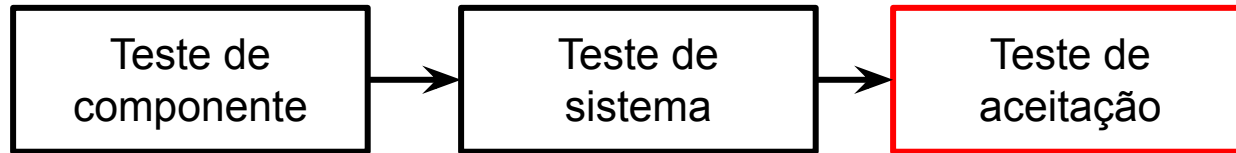
Testes de componente: Componentes do sistema (e.g., *funções*, *classes de objetos*) são testados de forma independente uns dos outros, pelas pessoas que o desenvolveram.

Verificação e Validação de software



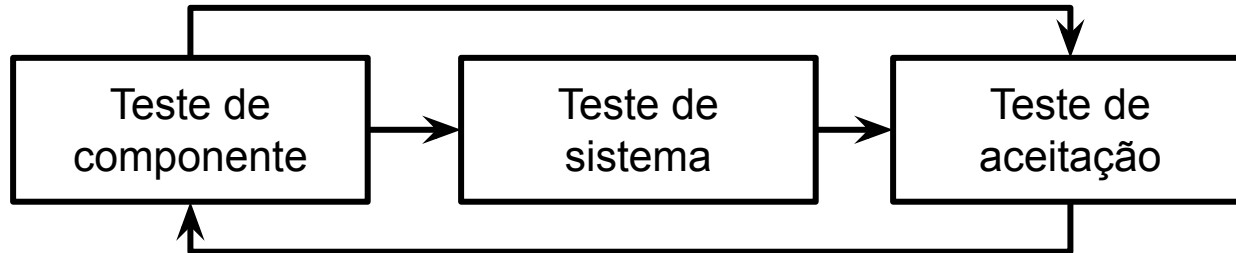
Testes de sistema: Componentes do sistema são integrados para criar um sistema completo.

Verificação e Validação de software



Testes de aceitação (ou alfa): O sistema é testado com dados fornecidos pelo cliente, e não com dados advindos de testes simulados.


Verificação e Validação de software



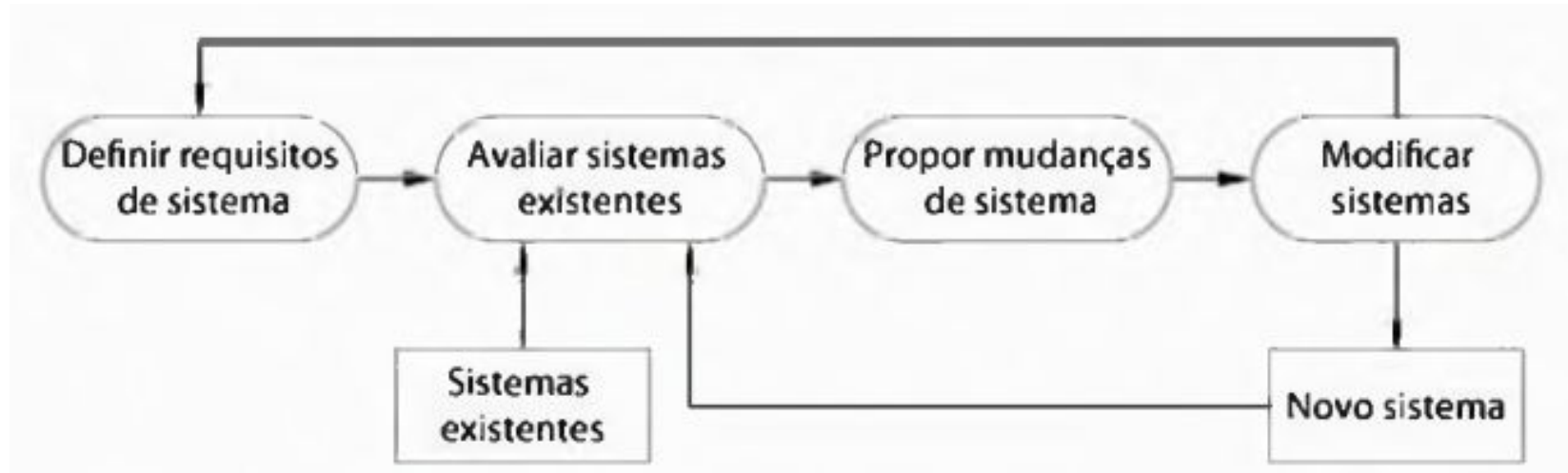
Se o sistema não for aceito pelo usuário, repetimos os testes.

Apesar da ilustração sequencial, **os processos de testes são intercalados.**

Processos de software

- **Especificação**
- **Desenvolvimento** (Projeto e Implementação)
- **Verificação e Validação**
- **Evolução** 

Evolução de software

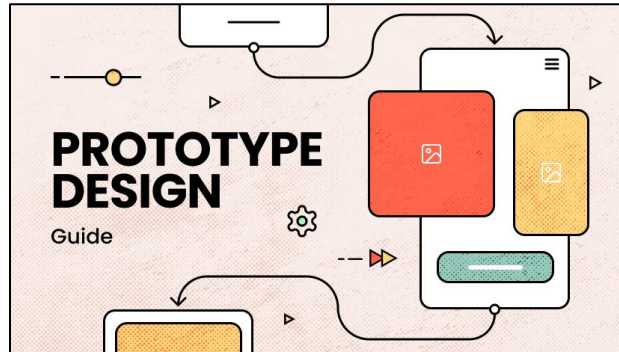


Evolução de software

- Evoluir um software significa dar manutenção nele
 - Ou seja, adicionar funcionalidades ou corrigir problemas
 - Isto é, evoluir significa, em muitos casos, que o esforço empregado em um sistema precisa ser refeito (**retrabalho**)
- **Abordagens para reduzir retrabalho**
 - **Prevenção de mudanças**
 - **Tolerância a mudanças**

Prevenção de mudanças

- Antecipar as mudanças possíveis antes que seja necessário qualquer retrabalho.
 - **Ex:** Testes com **protótipos** do sistema (*prototipação*)
 - **Objetivo:** refinar o sistema e seus requisitos antes de iniciar processos de desenvolvimentos (de alto custo)



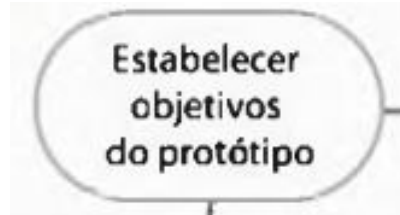
Prototipos de sistema (*prototipação*)

- Versão do sistema ou de parte dele, desenvolvida rapidamente
 - **Objetivo:**
 - Demonstrar conceitos
 - Verificar as necessidades do cliente
 - Verificar a viabilidade de algumas decisões de projeto
- Técnica de **prevenção de mudanças**
 - Usuários experimentem o sistema antes de sua entrega final
 - Menor número de mudanças de requisitos após entrega

Prototipos de sistema (*prototipação*)

- Ajudam a antecipar as mudanças que podem ser requisitadas, pois:
 - Eles ajudam na elicitac  o e valida  o de requisitos de sistema
 - Permitem estudar solu   es espec  ficas do software
 - **Ex:** *programa com GUI* ou *programa CLI*
 - Apoiam o projeto de interface de usu  rio
 - Permitem aos usu  rios ver qu  o bem o sistema d   suporte ao seu trabalho
 - Pode revelar erros e omiss  es nos requisitos propostos

Prototipos de sistema (*prototipação*)



Descreve quais são os objetivos da prototipação

Racional 01: um sistema possui muitas funcionalidades, e um protótipo não consegue não atender a todas simultaneamente

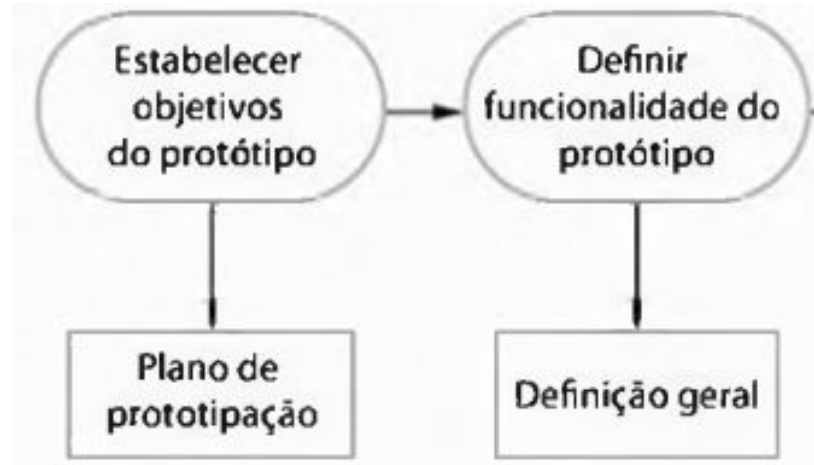
Racional 02: ao descrever os objetivos do protótipo, os usuários entendem melhor a sua função
(o que deve ser avaliado/testado através do protótipo)

Prototipos de sistema (*prototipação*)



Objetivos a serem alcançados com o prototipo

Prototipos de sistema (*prototipação*)



Descreve as funcionalidades que farão parte do protótipo, para reduzir custos e acelerar a entrega do sistema

Prototipos de sistema (*prototipação*)



Prototipos de sistema (*prototipação*)



Prototipos de sistema (*prototipação*)

- Desenvolvedores podem ser pressionados pelos gerentes para entregar protótipos descartáveis, gerando problemas
 - Dificuldade em atender aos requisitos não funcionais
 - Falta de documentação do protótipo
 - Degradação da estrutura do protótipo, causadas por mudanças com baixo grau de planejamento, durante o seu desenvolvimento
 - Baixo padrão de qualidade do protótipo (em relação aos padrões exigidos para o sistema final)

Protótipos de sistema (*prototipação*)

- Protótipos não precisam ser executáveis para serem úteis
 - **Ex:** Maquetes em papel da interface de usuário do sistema (RETTIG, 1994) podem ser eficazes para refinar o projeto de interface de usuário
 - Permitem simulações de uso através de **cenários de uso**
- **Cenários de uso:** descrições detalhadas de como um sistema será usado, que são utilizadas para validar requisitos e criar casos de teste

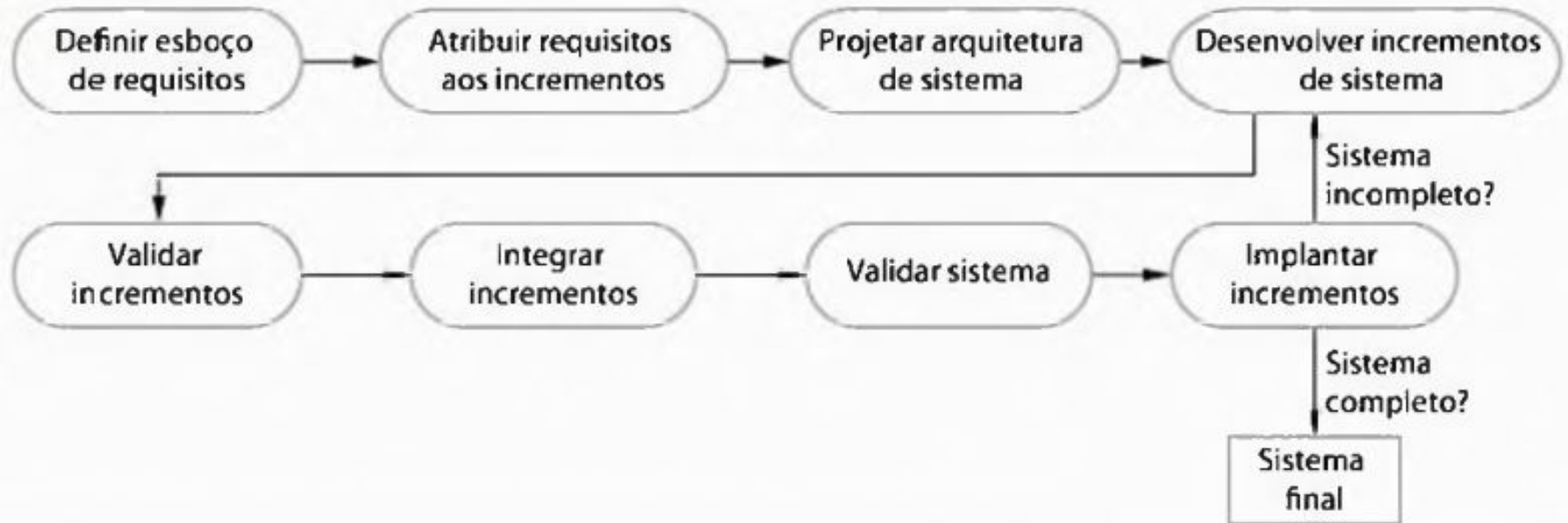
Tolerância a mudanças

- Processo projetado para permite que mudanças sejam acomodadas a um custo relativamente baixo
 - **Ex:** Desenvolvimento e entrega incrementais
 - **Objetivo:**
 - Alterações aplicadas em incrementos
 - Falhas em nas alterações propostas afetam apenas um incremento (parte do sistema)

Entrega incremental

- Incrementos do sistema são entregues aos clientes para comentários e experimentação.
 - Entrega antecipada de uma parte da funcionalidade do sistema
 - Ajuda os usuários a compreender suas necessidades para os incrementos posteriores
 - Técnica de **prevenção de mudanças e tolerância a mudanças**
 - Evita comprometimento prematuro com requisitos
 - Custo baixo de incorporação de mudanças nos incrementos

Entrega incremental



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.