




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 - PARTE 2 (Verificação / Validação e Evolução)

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

# 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

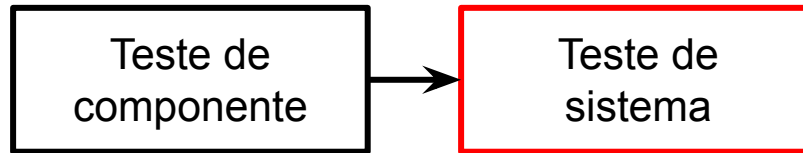
---

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

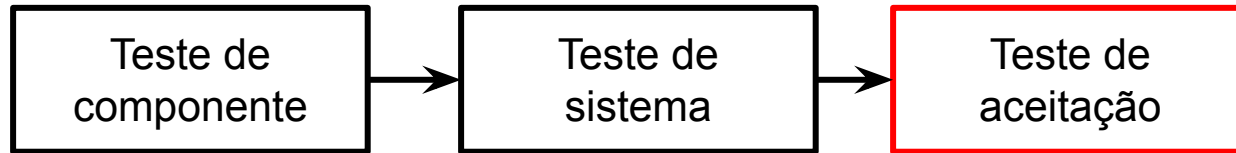
---



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

# Verificação e Validação de software

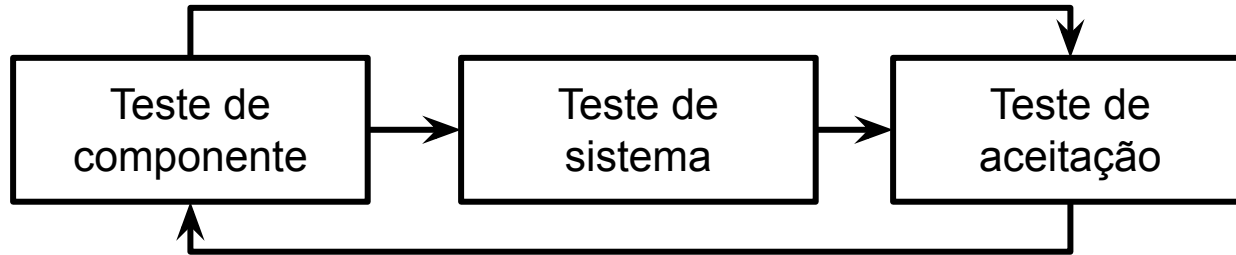
---



**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 testes são INTERCALADOS**

# Exercício

---

Considerando as atividades de projeto e implementação de software, marque a alternativa que contém **somente** as assertivas VERDADEIRAS.

I - Pelo processo geral de implementação de software, os programadores devem começar o desenvolvimento do sistema pelos componentes de mais fácil compreensão.

II - O debugging de código visa somente a localização dos defeitos deste sistema.

III - Para debugar um código, é necessário gerar hipóteses sobre o comportamento do programa e testar essas hipóteses para identificar a origem do defeito.

IV - Os defeitos de um sistema são identificados através de testes de defeitos.

- ☐ Somente I.
- ☐ Somente II.
- ☐ Somente III.
- ☐ Somente IV.
- ☐ Somente III e IV.



# Exercício

Considerando as atividades de projeto e implementação de software, marque a alternativa que contém **somente** as assertivas VERDADEIRAS.

- I - Pelo processo geral de implementação de software, os programadores devem começar o desenvolvimento do sistema pelos componentes de mais fácil compreensão. **F**
- II - O debugging de código visa somente a localização dos defeitos deste sistema.
- III - Para debugar um código, é necessário gerar hipóteses sobre o comportamento do programa e testar essas hipóteses para identificar a origem do defeito.
- IV - Os defeitos de um sistema são identificados através de testes de defeitos.

- ☐ Somente I.
- ☐ Somente II.
- ☐ Somente III.
- ☐ Somente IV.
- ☐ Somente III e IV.

# Exercício

Considerando as atividades de projeto e implementação de software, marque a alternativa que contém **somente** as assertivas VERDADEIRAS.

I - Pelo processo geral de implementação de software, os programadores devem começar o desenvolvimento do sistema pelos componentes de mais fácil compreensão. **F**

II - O debugging de código visa somente a localização dos defeitos deste sistema. **F**

III - Para debugar um código, é necessário gerar hipóteses sobre o comportamento do programa e testar essas hipóteses para identificar a origem do defeito.

IV - Os defeitos de um sistema são identificados através de testes de defeitos.

☐ Somente I.

☐ Somente II.

☐ Somente III.

☐ Somente IV.

☐ Somente III e IV.

# Exercício

Considerando as atividades de projeto e implementação de software, marque a alternativa que contém **somente** as assertivas VERDADEIRAS.

I - Pelo processo geral de implementação de software, os programadores devem começar o desenvolvimento do sistema pelos componentes de mais fácil compreensão. **F**

II - O debugging de código visa somente a localização dos defeitos deste sistema. **F**

III - Para debugar um código, é necessário gerar hipóteses sobre o comportamento do programa e testar essas hipóteses para identificar a origem do defeito. **V**

IV - Os defeitos de um sistema são identificados através de testes de defeitos.

- ☐ Somente I.
- ☐ Somente II.
- ☐ Somente III.
- ☐ Somente IV.
- ☐ Somente III e IV.

# Exercício


Considerando as atividades de projeto e implementação de software, marque a alternativa que contém **somente** as assertivas VERDADEIRAS.

- I - Pelo processo geral de implementação de software, os programadores devem começar o desenvolvimento do sistema pelos componentes de mais fácil compreensão. **F**
- II - O debugging de código visa somente a localização dos defeitos deste sistema. **F**
- III - Para debugar um código, é necessário gerar hipóteses sobre o comportamento do programa e testar essas hipóteses para identificar a origem do defeito. **V**
- IV - Os defeitos de um sistema são identificados através de testes de defeitos. **V**

- ☐ Somente I.
- ☐ Somente II.
- ☐ Somente III.
- ☐ Somente IV.
- ☒ Somente III e IV.

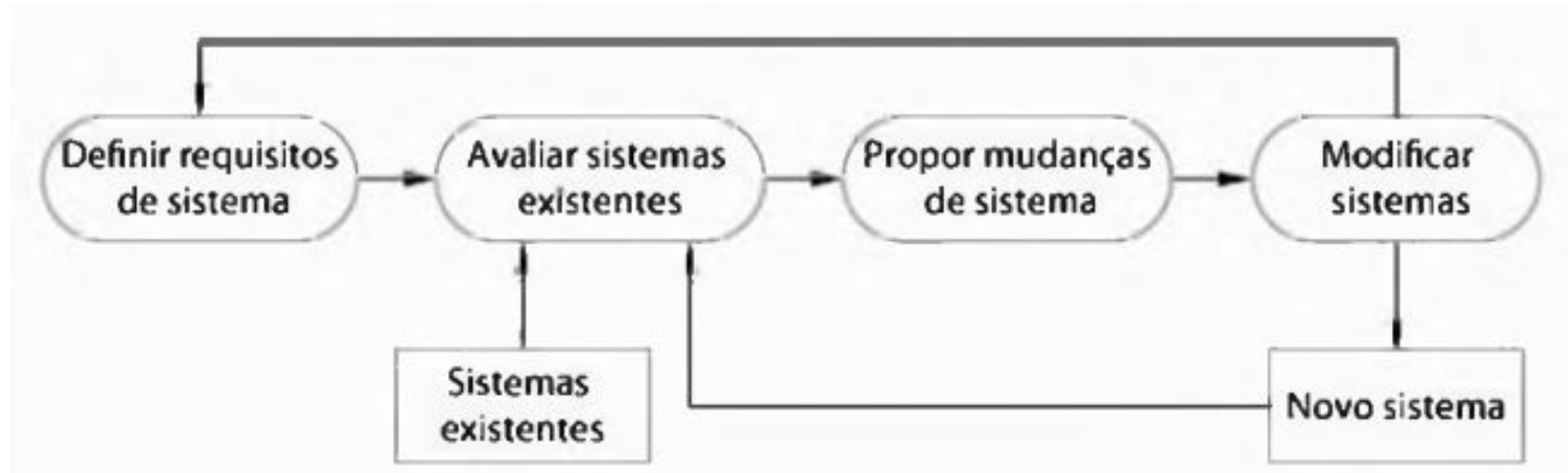
# Processos de software

---

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

# Evolução de software

---



# Evolução de software

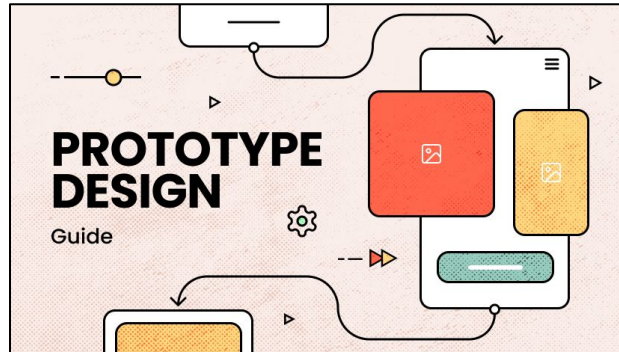
---

- 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

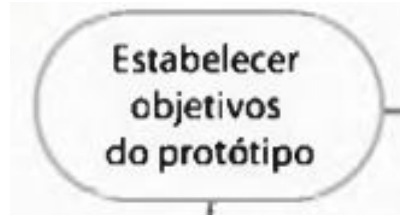
# Prototipos de sistema (*prototipação*)

---

- 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 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*)

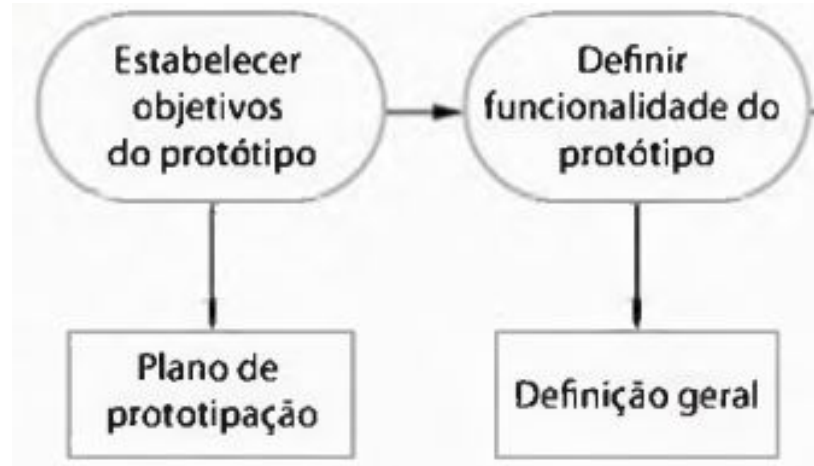
---



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*)

---



# Exercício

---

Considerando a prototipação de sistemas, marque a alternativa que contém **somente** as assertivas VERDADEIRAS.

I - Protótipos ajudam a antecipar mudanças em sistemas, pois eles ajudam na elicitacão e validação de requisitos.

II - Protótipos podem apoiar o projeto de interface de usuário.

III - Protótipos permitem que os usuários avaliem o sistema conforme este é desenvolvido.

IV - Protótipos podem revelar erros e omissões nos requisitos propostos.

- ☐ Todas as assertivas são verdadeiras.
- ☐ Somente I e IV.
- ☐ Somente II e III.
- ☐ Somente II, III e IV.
- ☐ Nenhuma das alternativas anteriores.



# Exercício

---

Considerando a prototipação de sistemas, marque a alternativa que contém **somente** as assertivas VERDADEIRAS.

I - Protótipos ajudam a antecipar mudanças em sistemas, pois eles ajudam na elicitação e validação de requisitos. **V**

II - Protótipos podem apoiar o projeto de interface de usuário.

III - Protótipos permitem que os usuários avaliem o sistema conforme este é desenvolvido.

IV - Protótipos podem revelar erros e omissões nos requisitos propostos.

- ☐ Todas as assertivas são verdadeiras.
- ☐ Somente I e IV.
- ☐ Somente II e III.
- ☐ Somente II, III e IV.
- ☐ Nenhuma das alternativas anteriores.

# Exercício

---

Considerando a prototipação de sistemas, marque a alternativa que contém **somente** as assertivas VERDADEIRAS.

I - Protótipos ajudam a antecipar mudanças em sistemas, pois eles ajudam na elicitação e validação de requisitos. **V**

II - Protótipos podem apoiar o projeto de interface de usuário. **V**

III - Protótipos permitem que os usuários avaliem o sistema conforme este é desenvolvido.

IV - Protótipos podem revelar erros e omissões nos requisitos propostos.

- ☐ Todas as assertivas são verdadeiras.
- ☐ Somente I e IV.
- ☐ Somente II e III.
- ☐ Somente II, III e IV.
- ☐ Nenhuma das alternativas anteriores.

# Exercício

---

Considerando a prototipação de sistemas, marque a alternativa que contém **somente** as assertivas VERDADEIRAS.

I - Protótipos ajudam a antecipar mudanças em sistemas, pois eles ajudam na elicitação e validação de requisitos. **V**

II - Protótipos podem apoiar o projeto de interface de usuário. **V**

III - Protótipos permitem que os usuários avaliem o sistema conforme este é desenvolvido. **V**

IV - Protótipos podem revelar erros e omissões nos requisitos propostos.

- ☐ Todas as assertivas são verdadeiras.
- ☐ Somente I e IV.
- ☐ Somente II e III.
- ☐ Somente II, III e IV.
- ☐ Nenhuma das alternativas anteriores.

# Exercício

Considerando a prototipação de sistemas, marque a alternativa que contém **somente** as assertivas VERDADEIRAS.

I - Protótipos ajudam a antecipar mudanças em sistemas, pois eles ajudam na elicitação e validação de requisitos. **V**

II - Protótipos podem apoiar o projeto de interface de usuário. **V**

III - Protótipos permitem que os usuários avaliem o sistema conforme este é desenvolvido. **V**

IV - Protótipos podem revelar erros e omissões nos requisitos propostos. **V**

- ☒ Todas as assertivas são verdadeiras.
- ☐ Somente I e IV.
- ☐ Somente II e III.
- ☐ Somente II, III e IV.
- ☐ Nenhuma das alternativas anteriores.

# 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 nas alterações propostas afetam apenas um incremento (parte do sistema)

# Entrega incremental

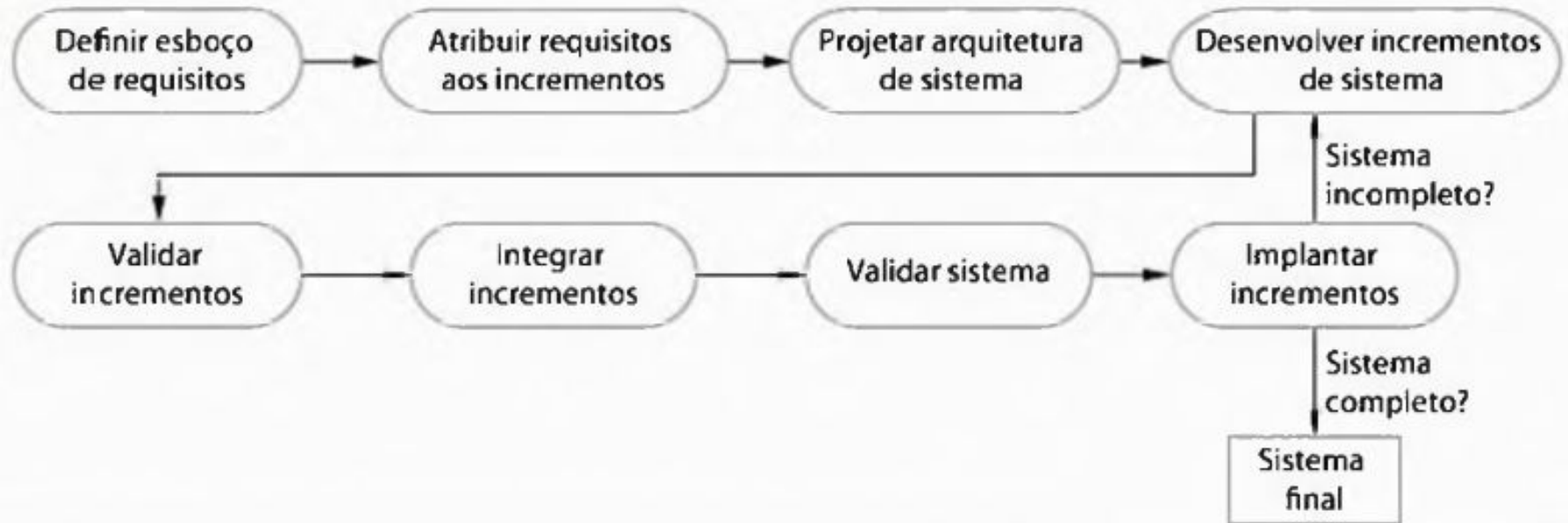
---

- 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

---



# Atividade em sala

---

- Em grupo, utilizem o **PlantUML** para construir o planejamento de execução do projeto, incluindo:
  - **Previsão para execução de cada atividade** (cronograma)
  - **Prazos** (*deadlines*)
- **Exemplos e Tutoriais do PlantUML:**
  - [https://github.com/andre-romano/aulas/tree/master/eng\\_soft1/plantuml](https://github.com/andre-romano/aulas/tree/master/eng_soft1/plantuml)
  - [https://www.youtube.com/watch?v=WSC1K\\_rDf2w](https://www.youtube.com/watch?v=WSC1K_rDf2w)

# 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.