Modelo Cascata

Prof. Dr. William Simão de Deus

william.deus@ifpr.edu.br
Instituto Federal do Paraná (IFPR)
Campus Pinhais
Gestão da Tecnologia da Informação – 2025.01

Sumário

Origem e Definição

Segundo (WAZLAWICK, 2019), o modelo cascata:

- Surgiu nos anos 1970
- Modelo de desenvolvimento detalhado e previsível focado nos grandes sistemas militares
- Análise e projeto detalhados antes da codificação
- Foco em aproximar o código dos requisitos do cliente

Visões sobre o modelo Cascata

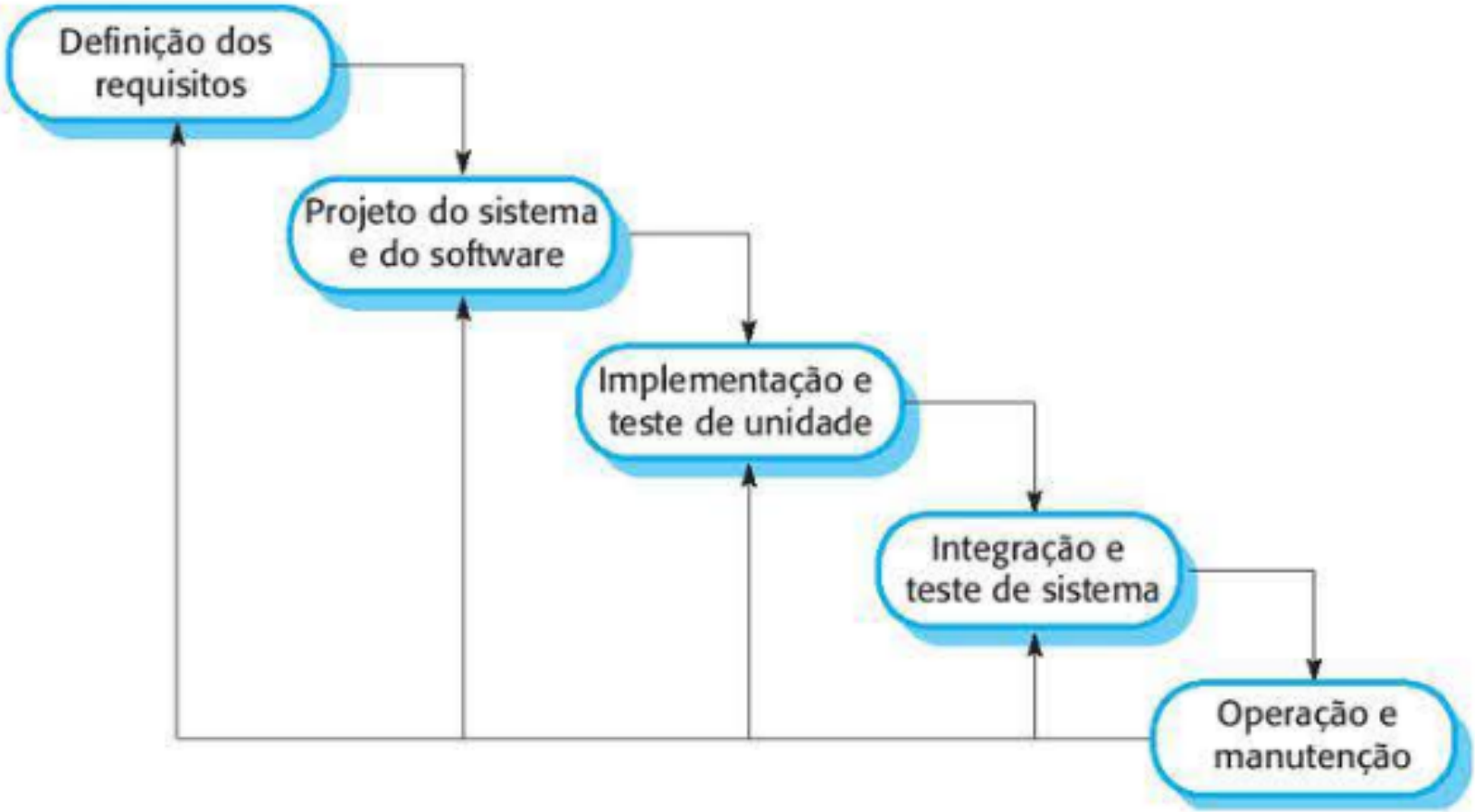

Figura: (SOMMERVILLE, 2011)

Visões sobre o modelo Cascata

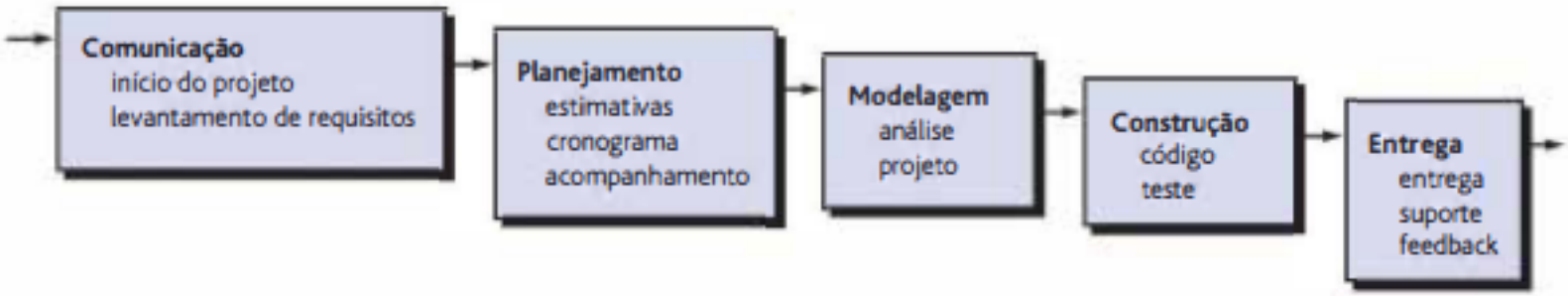

FIGURA 4.1 O modelo cascata.

Figura: (PRESSMAN, 2005)

- Os **serviços**, **restrições** e **metas** do sistema são definidos
- O processo envolve a consulta aos usuários para estabelecer os requisitos
- Gera-se uma espécie de **especificação detalhada** do sistema

2. Projeto do Sistema e do Software - (SOMMERVILLE, 2011)

- Os requisitos são divididos entre **hardware** e **software**
- No projeto de software, são identificadas as **abstrações fundamentais** e seus relacionamentos

- O projeto do software é traduzido em um **conjunto de programas** ou unidades
- O **teste de unidade** verifica cada unidade individualmente
- O objetivo é garantir que cada parte satisfaça sua especificação

- As unidades são **integradas** e testadas como um **sistema completo**.
- O teste visa garantir que todos os requisitos de software foram cumpridos
- Após o teste, o sistema é **entregue ao cliente**

- Esta é a **fase mais longa** do ciclo de vida do software.
- O sistema é instalado, colocado em uso e corrigido.
- A manutenção inclui:
  - Corrigir erros não descobertos
  - Melhorar a implementação
  - Aperfeiçoar os serviços conforme novos requisitos

Pontos de atenção

- Na prática, cada fase do modelo em cascata gera uma série de documentações
- Esses documentos podem ser atualizados de acordo com o andamento do projetos
- Alguns projetos são *congelados* para evitar aterações e o impacto dessas mudanças
- O modelo em cascata não prevê validação pelo cliente em todas as fases do processo de desenvolvimento do software

Lidando com a instabilidade

- No desenvolvimento de **software**, é comum que exista grande sobreposição e feedbacks entre as fases
  - Durante o desenvolvimento, são identificados problemas com os requisitos, por exemplo

Lidando com a instabilidade

- No desenvolvimento de **software**, é comum que exista grande sobreposição e feedbacks entre as fases
  - Durante o desenvolvimento, são identificados problemas com os requisitos, por exemplo
- No cascata, esse tipo de problema gera uma atualização na documentação para refletir as mudanças do sistema
  - No caso de mudanças, as partes precisam ser comunicadas e acordar com a atualização proposta

Lidando com a instabilidade

- Na prática, tanto clientes quanto desenvolvedores podem congelar o desenvolvimento do projeto
- Isso abre portas ao desenvolvimento de sistemas mal estruturados ou com contornados com soluções inadequadas

Benefícios	Desafios
• Estrutura organizada e bem definida • Foco em um planejamento detalhado antes da codificação • Facilita o controle do progresso, com fases bem estabelecidas	• Projetos raramente seguem o fluxo linear proposto, e mudanças podem causar confusão • Dificuldade em acomodar a incerteza inicial e necessidades não completamente claras • Longo tempo até o cliente validar uma versão funcional, o que pode aumentar o risco de erros graves não detectados

Tabela: Benefícios e Desafios do Modelo Cascata (PRESSMAN, 2005)

Pontos de atenção

- Sistemas embarcados (software interage com o hardware), sistemas críticos e grandes sistemas de software são os tipos de projetos mais adequados para a aplicação do modelo cascata
- Ambientes instáveis, com mudança de requisitos e comunicação informação, não são adequados para o cascata

Críticas ao modelo - (PRESSMAN, 2005)

- Projetos reais raramente seguem o uxo sequencial proposto pelo modelo

Críticas ao modelo - (PRESSMAN, 2005)

- Projetos reais raramente seguem o uxo sequencial proposto pelo modelo
- Com frequência, é difícil para o cliente estabelecer explicitamente todas as necessidades no início da maioria dos projetos

Críticas ao modelo - (PRESSMAN, 2005)

- Projetos reais raramente seguem o uxo sequencial proposto pelo modelo
- Com frequência, é difícil para o cliente estabelecer explicitamente todas as necessidades no início da maioria dos projetos
- O cliente deve ter paciência. Uma versão operacional do(s) programa(s) não estará disponível antes de estarmos próximos ao final do projeto

Críticas ao modelo - (PRESSMAN, 2005)

- Projetos reais raramente seguem o uxo sequencial proposto pelo modelo
- Com frequência, é difícil para o cliente estabelecer explicitamente todas as necessidades no início da maioria dos projetos
- O cliente deve ter paciência. Uma versão operacional do(s) programa(s) não estará disponível antes de estarmos próximos ao final do projeto
- Erros graves podem não ser detectados até o programa operacional ser revisto.

Críticas ao modelo - (WAZLAWICK, 2019)

- A atribuição do modelo é dada a Royce, que usou o termo *Waterfall* para designá-lo
- Ironicamente, Royce apresentava justamente esse modelo como algo que não deveria ser feito
- Tendo em vista que muitas funcionalidades seriam testadas e experimentadas pela primeira vez na fase de testes

- O que acontece se um erro grave for detectado tardiamente no Modelo Cascata?

Cascata

- O que acontece se um erro grave for detectado tardiamente no Modelo Cascata?
  - Retrabalho intenso

- O que acontece se um erro grave for detectado tardiamente no Modelo Cascata?
  - Retrabalho intenso
  - Possíveis atrasos

- O que acontece se um erro grave for detectado tardiamente no Modelo Cascata?
  - Retrabalho intenso
  - Possíveis atrasos
  - Aumento de custos

- O que acontece se um erro grave for detectado tardiamente no Modelo Cascata?
  - Retrabalho intenso
  - Possíveis atrasos
  - Aumento de custos
  - Impacto na qualidade

Dúvidas?

Referências

PRESSMAN, R. S. Software engineering: a practitioner's approach. *Pressman and Associates*, 2005.

SOMMERVILLE, I. *Engenharia de Software - 9 ed.* [S.l.]: Pearson, 2011.

WAZLAWICK, R. *Engenharia de software: conceitos e práticas*. [S.l.]: Elsevier Editora Ltda., 2019.