Projeto de Desenvolvimento de Software

Modelos de Processo Prescritivos

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

Msc. Eliezio Soares
eliezio.soares@ifrn.edu.br
http://docente.ifrn.edu.br/elieziosoares

Avisos

- Atividade Avaliativa: 30/06/2015
  - Composição da Nota do 1º bimestre:
    - 2.0 pontos - Atividade 1: Entrevista
    - 2.0 pontos – Atividade 2: Sábado Letivo
    - 6.0 pontos - Prova

Objetivos

- Conceituar o que são modelos de processo prescritivos;
- Exemplificar um antimodelo;
- Explorar o Modelo Cascata;
- Explorar o Modelo Espiral;

Modelos Prescritivos

- Consideremos que existem duas grandes classes de modelos:
  - Modelos Ágeis
  - Modelos Prescritivos:

> "São modelos que definem um conjunto distinto de atividades, tarefas, marcos e produtos de trabalho que são necessários para fazer engenharia de software com alta qualidade."
>
> (Pressman. 2005)

Processos "que se baseiam em uma descrição de como as atividades são feitas."

(Wazlawick, Raul Sidnei. 2013)

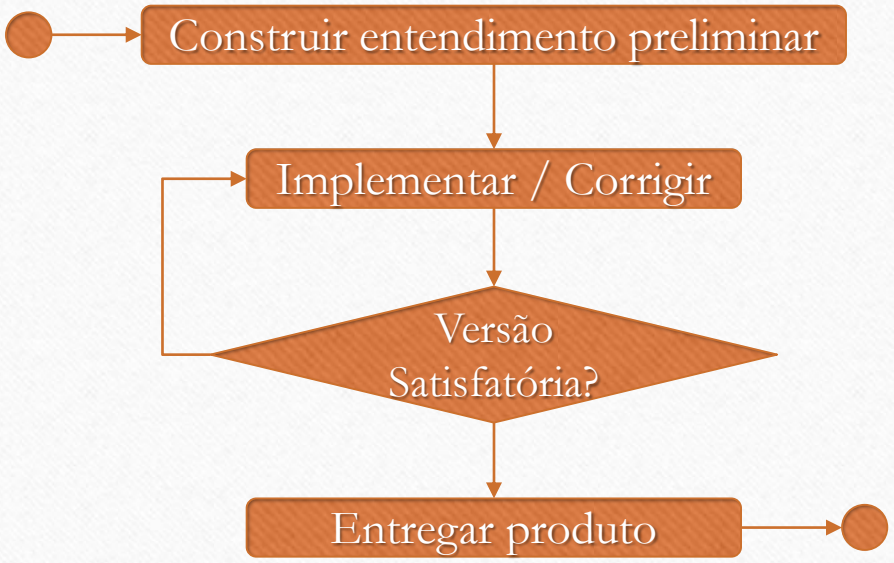
Principais Modelos Prescritivos

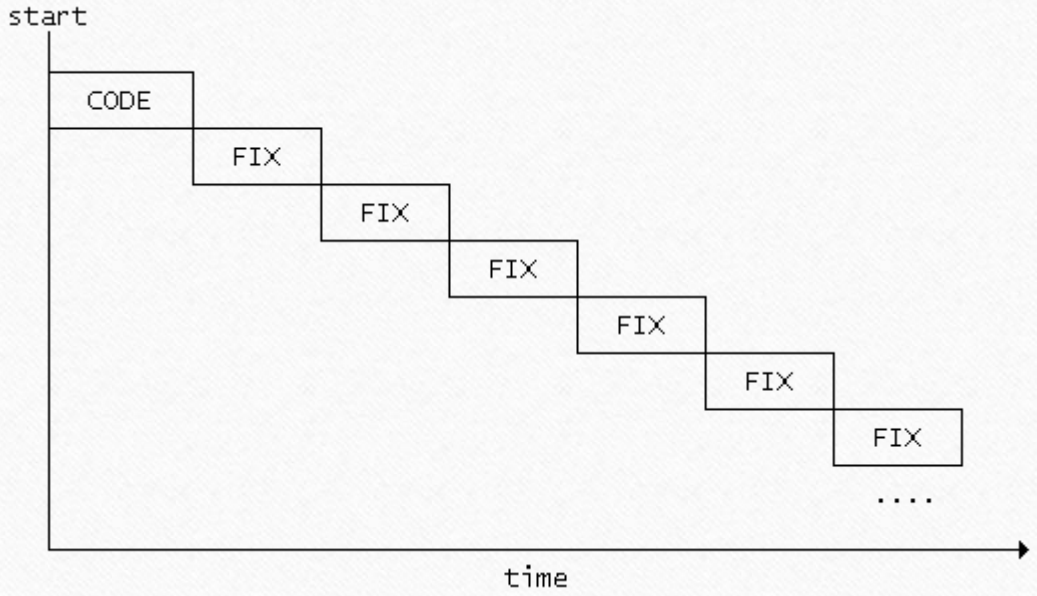
- Waterfall ou Cascata
  - Possui fases bem definidas e sequenciais.

- Espiral
  - Possui ciclos de prototipação visando a redução de riscos de projeto.

Modelo Mais Famoso: Codificar e Consertar (*Code and Fix*)

- Filosofia Simples:





Codificar e Consertar (*Code and Fix*)

- Em Resumo:
  1. Construir com o cliente um entendimento preliminar sobre o sistema que deve ser desenvolvido.
  2. Implementar uma primeira versão desse sistema.
  3. Interagir com o cliente de forma a corrigir a versão preliminar até que esta satisfaça o cliente.
  4. Fazer testes e corrigir os erros inevitáveis.
  5. Entregar o produto.

Codificar e Consertar (*Code and Fix*)
O antimodelo

- *Code and Fix* **não** é um modelo de processo.
  - Não há previsibilidade em relação às atividades;
  - Não há previsibilidade em relação aos resultados obtidos;

- Segundo (McConnell, 1996) *Code and Fix* deve ser entendido mais como uma maneira de pressionar os programadores do que como um processo organizado.

(Des)Vantagens do *Code and Fix*

1. Não há tempo em documentação, planejamento ou projeto: “Direto ao código!”
2. O progresso é facilmente visível a medida que o programa vai ficando “pronto”.
3. Não há necessidade de treinamento ou conhecimentos especiais. Qualquer programador pode desenvolver software com esse modelo de ciclo de vida.

Desvantagens do *Code and Fix*

1. É muito difícil avaliar a qualidade e os riscos do projeto.
2. Se no meio do projeto a equipe descobrir que as decisões arquiteturais estavam erradas, não há solução, a não ser começar tudo de novo.

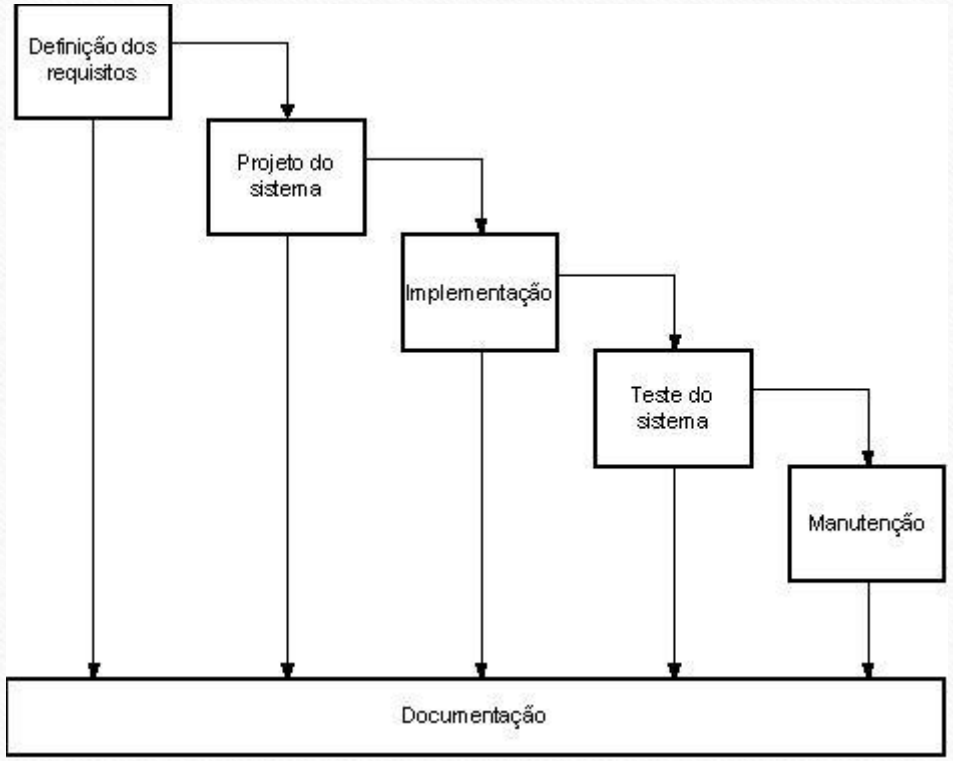
LEI DA COMPLEXIDADE CRESCENTE – A medida que mudanças são introduzidas no software, as interações entre elementos fazem que a entropia interna aumente, ou seja, ocorre um crescimento cada vez mais desestruturado.

(Lehman M.M., 1974)

Modelo Cascata

- “O avô de todos os modelos”
- Começou a ser definido nos anos 1970;
- Filosofia:
  - BDUF – *Big Design Up Front:*
    - Antes de linhas de código, é preciso fazer um trabalho detalhado de análise e projeto, de forma que, quando o código for efetivamente produzido, esteja o mais próximo possível dos requisitos do cliente.
    - Antes de avançar à próxima fase deve haver uma revisão da fase anterior.

Disciplinas



Vantagens do Modelo Cascata

- Fases bem definidas ajuda a detectar erros cedo (prejuízo menor);
- Busca promover a estabilidade dos requisitos; assim, o projeto só segue em frente quando os requisitos são aceitos;
- Funciona bem com requisitos bem conhecidos e estáveis;
- Funciona bem quando a preocupação com a qualidade está acima das preocupações com custo ou tempo de desenvolvimento;
- É adequado para equipes tecnicamente fracas ou inexperientes, pois fornece uma estrutura sólida ao projeto, direcionando todos os esforços.

Desvantagens do Modelo Cascata

- Verificar código é fácil. Mas como verificar modelos e projetos?
- Não produz resultados tangíveis até a fase de codificação (na perspectiva do cliente);
- É difícil estabelecer requisitos completos antes de começar a codificar.
  - Desenvolvedores sempre reclamam que usuários não sabem expressar o que precisam.
- Não há flexibilidade com requisitos.

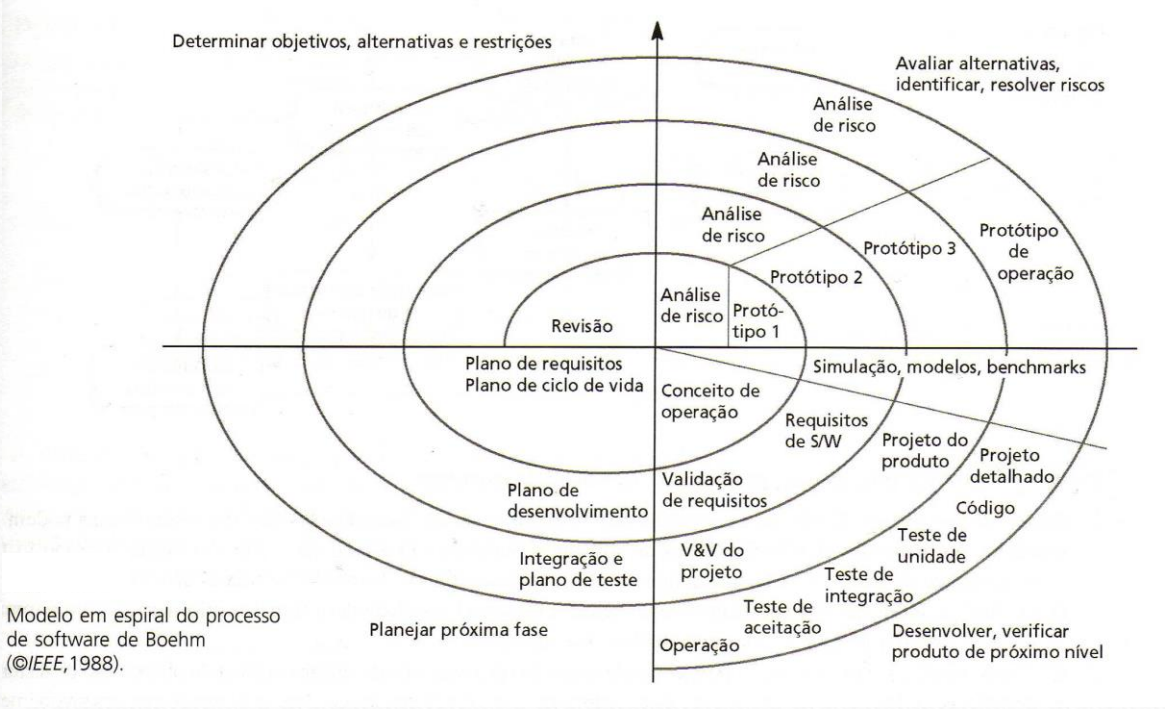
Modelo Cascata

- Mencionado na literatura a primeira vez por Royce (1970) como modelo que não deveria ser seguido.
  - Embora acreditasse na filosofia organizada, julgava sua implementação arriscada pois vários aspectos do software seriam experimentados somente na fase de testes.

Modelo Espiral

- Proposto por Boehm (1986);
- Orientado a redução de riscos;
- Baseado na realização de ciclos iterativos;

Modelo Espiral



Modelo em espiral do processo de software de Boehm (©*IEEE*,1988).

"A ideia desse ciclo é iniciar com **miniprojetos**, abordando os principais **riscos**, e então expandir o projeto através da construção de **protótipos**, **testes** e **replanejamento**, de forma a abarcar os **riscos** identificados. Após a equipe ter adquirido um conhecimento mais completo dos potenciais problemas com o sistema, passará a desenvolver um ciclo final semelhante ao do modelo cascata."

(Wazlawick, Raul Sidnei. 2013)

Passos do Modelo Espiral

1. Determinar inicialmente os objetivos, alternativas e restrições relacionadas à iteração que vai se iniciar.
2. Identificar e resolver riscos relacionados à iteração em andamento.
3. Avaliar as alternativas disponíveis.
   - Podem ser utilizados protótipos para verificar a viabilidade de diferentes alternativas.
4. Desenvolver os artefatos relacionados a essa iteração e certificar-se de que estão corretos.
5. Planejar a próxima iteração.
6. Obter concordância em relação à abordagem para a próxima iteração, caso se resolva realizar uma.

Vantagens do Modelo Espiral

- As primeiras iterações são as mais baratas no que diz respeito a tempo e recursos.
- A escolha dos riscos a serem explorados é feita em função das necessidades do projeto.
- A medida que os custos aumentam, os riscos diminuem (um sonho!).

Desvantagens do Modelo Espiral

- Este modelo não fornece indicações suficientes sobre quantidade de trabalho esperada em cada ciclo.
- O tempo de desenvolvimento (prazo) se torna imprevisível.
- Esse modelo possui movimentação entra fases complexa o que torna complexo o gerenciamento do projeto.

Indicações e Contra Indicações

- Modelo Cascata
  - Indicado para:
    - Projetos muito bem conhecidos, com requisitos claros.
  - Não indicado para:
    - Projetos complexos ou com requisitos pouco conhecidos.
- Modelo Espiral
  - Indicado para:
    - Projetos complexos, com alto risco e requisitos pouco conhecidos (P&D).
  - Não indicado para:
    - Projetos de pequeno e médio porte ou com requisitos bem conhecidos.

Referências

- Wazlawick, Raul Sidnei. Engenharia de software: conceitos e práticas. Rio de Janeiro: Elsevier, 2013.