Engenharia de Software I
04 – Processos e Ciclo de vida

Márcio Daniel Puntel
marcio@puntel.org

Processos de Software

- Especificação
  - Engenharia de Sistema: solução para o problema.
  - Análise de Requisitos: necessidades do software a ser implementado – resulta em um documento.
  - Especificação de Sistema: descrição funcional do sistema.

Processos de Software

- Projeto
  - Projeto Arquitetural: modelo conceitual para o sistema, composto de módulos mais ou menos independentes.
  - Projeto de Interface: onde cada módulo tem sua interface de comunicação estudada e definida.
  - Projeto Detalhado: onde os módulos em si são definidos, e possivelmente traduzidos para pseudocódigo.

Processos de Software

- Implementação
  - Codificação: implementação em uma linguagem de computador.
- Validação
  - Teste de Unidade e Módulo: realização de testes para verificar a inexistência de erros e comportamento adequado ao sistema.
  - Integração: a reunião dos diferentes módulos em um produto de software homogêneo, e a verificação da interação entre estes quando operando em conjunto.
- Manutenção e Evolução
  - Nesta fase, o software em geral entra em um ciclo iterativo que abrange todas as fases anteriores.

Modelos de Ciclo de vida

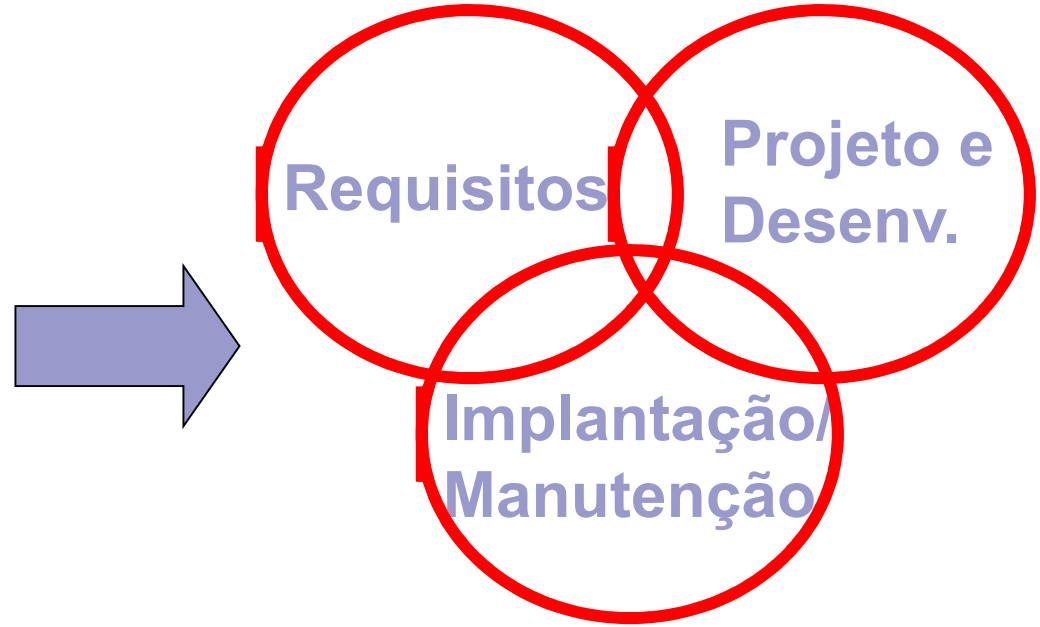
- Projetos são empreendimentos únicos e portanto envolvem um grau de incerteza.
- Organizações dividem projetos em fases de forma a garantir um melhor controle e encadeamento com as operações correntes da empresa.
- O conjunto das fases de um projeto é conhecido como ciclo de vida do projeto.

Seqüência das fases

- A sequência de fases normalmente envolve alguma forma de transferência de informação. Exemplo: de requisitos para projeto ou de projeto para construção.
- Produtos das fases precedentes são usualmente aprovados **antes** do início da fase seguinte.
- “Fast tracking”: fases correndo em paralelo com um risco tolerável.

Ciclo de Vida do Software

Fases
(macro-etapas)



03 Macro-Etapas

- Requisitos: “O que” deve ser desenvolvido e “Como” os processos e dados são utilizados.

Analista + Cliente/Usuário

- Projeto e Desenvolvimento:

Analista + Programador => Cliente/Usuário

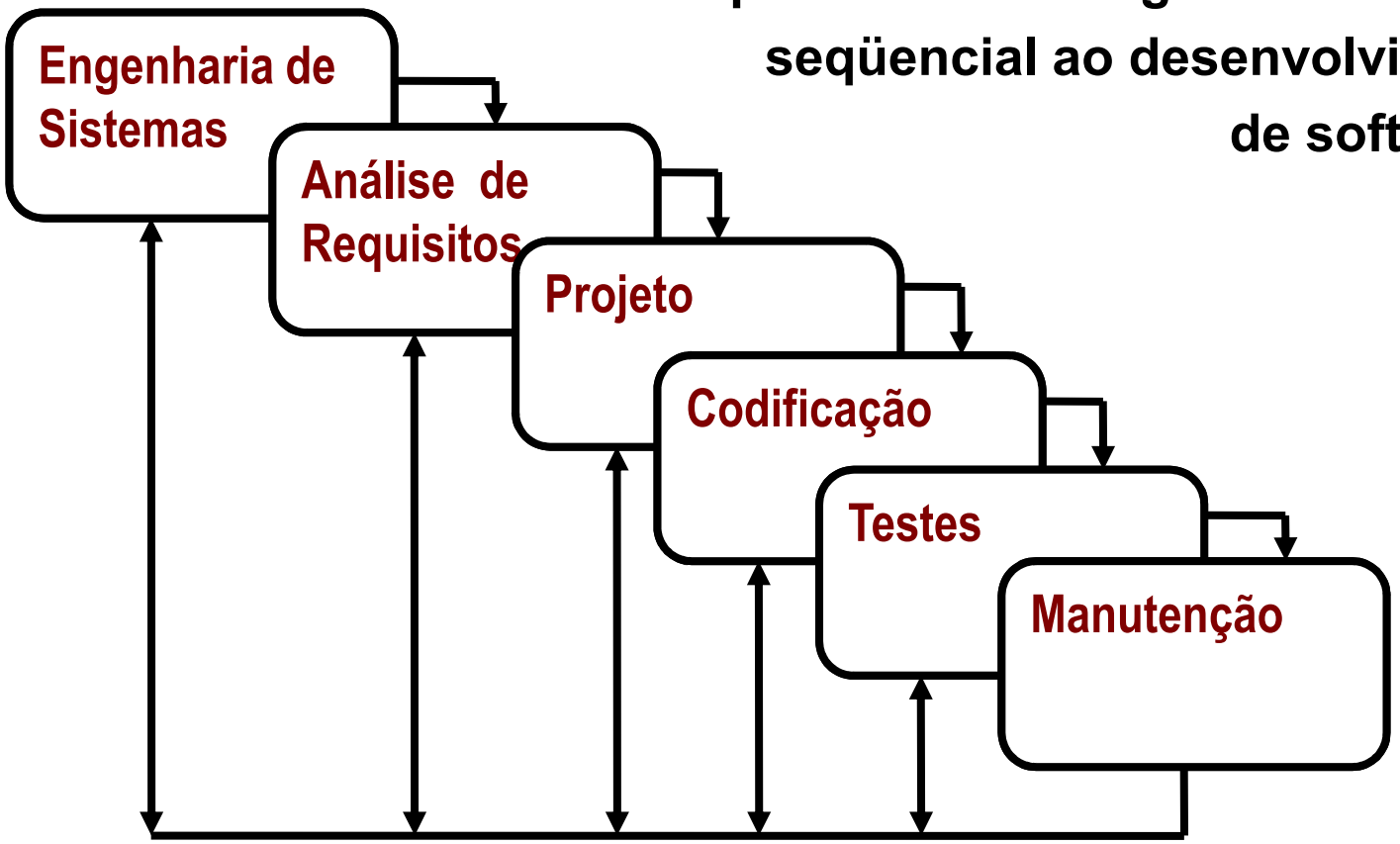
- Implantação: Resistência
- Manutenção: Vida útil (correção, adaptação, incorporação de melhoramento funcional)

Modelos de Ciclo de vida

- Modelos de ciclos de vida definem:
  - que parte do trabalho técnico que deve ser feita em cada fase
  - quem deve estar envolvido em cada fase
  - como deve ser feito o trabalho em cada fase

Modelo Cascata

- **Requer uma abordagem sistemática seqüencial ao desenvolvimento de software.**



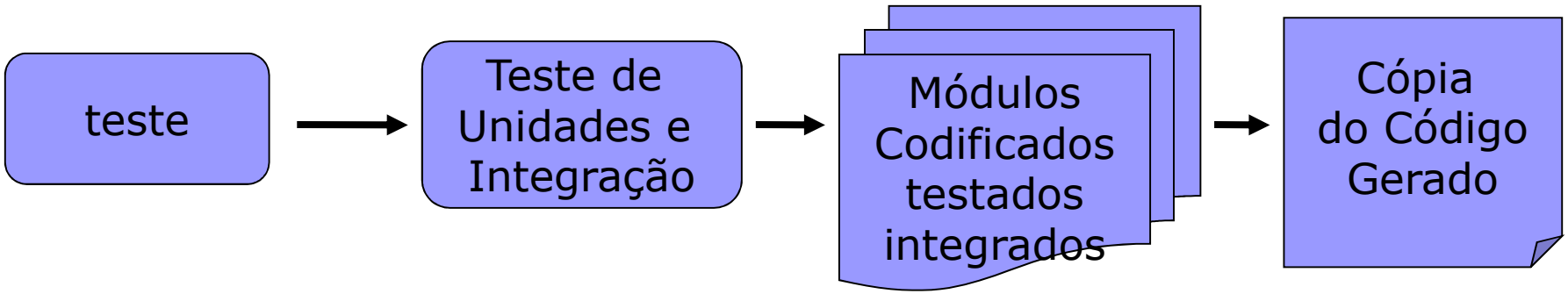
Modelo Cascata - Características

- Uma etapa termina antes da próxima começar
- Cliente afastado da fase de implementação (funcionalidade do sistema separada do projeto); "o que – como fazer"
- Foi e ainda tem sido usado para indicar as atividades de projeto em vários contextos

Modelo Cascata

Um exemplo do uso deste modelo (Departamento de defesa dos EUA)

- Gerentes de projeto podiam utilizar o modelo para estimar quanto faltava para o término do projeto.
- Marcos e Produtos estavam associados a cada atividade do processo.



Modelo Cascata

- Ajuda os desenvolvedores a descrever o que precisam fazer (útil);
- Ajuda a explicar o processo de desenvolvimento aos clientes (simples e fácil);
- Explicita os produtos intermediários para começar próximo estágio;
- Seu enfoque está nos documentos e artefatos (requisitos, projetos, códigos);

Modelo Cascata

- Maior problema: não reflete efetivamente como o código é desenvolvido
- Frequentemente usado na solução de problemas nunca resolvidos antes

Modelo Cascata

- Usuários e Desenvolvedores não irão conhecer todos os requisitos em uma única etapa (todos os fatores que influenciarão no resultado desejado)
- Usar muito do tempo gasto na compreensão dos itens e processos afetados pelo sistema e seu software;

Modelo Cascata - Dicas

- Controlar o processo real de desenvolvimento;
- Não passar muitas vezes de uma etapa para a outra e ficar voltando para a anterior sucessivamente enquanto se esforçam para adquirir conhecimento sobre o problema e como chegar a solução proposta.

Modelo Cascata - Desvantagens

- Não mostra detalhes de como cada fase transforma um artefato em outro, por exemplo: Como transformar os requisitos em projeto;
- Não trata sobre as mudanças nos produtos e atividades que provavelmente ocorrerão durante o desenvolvimento.

Modelo Cascata - Desvantagens

- Exemplos:
  - quando os requisitos são modificados durante a programação, as mudanças subsequentes no projeto e no código não são indicadas.
  - Não informa nada sobre as atividades de “ida e volta” que levam a criação do produto final.

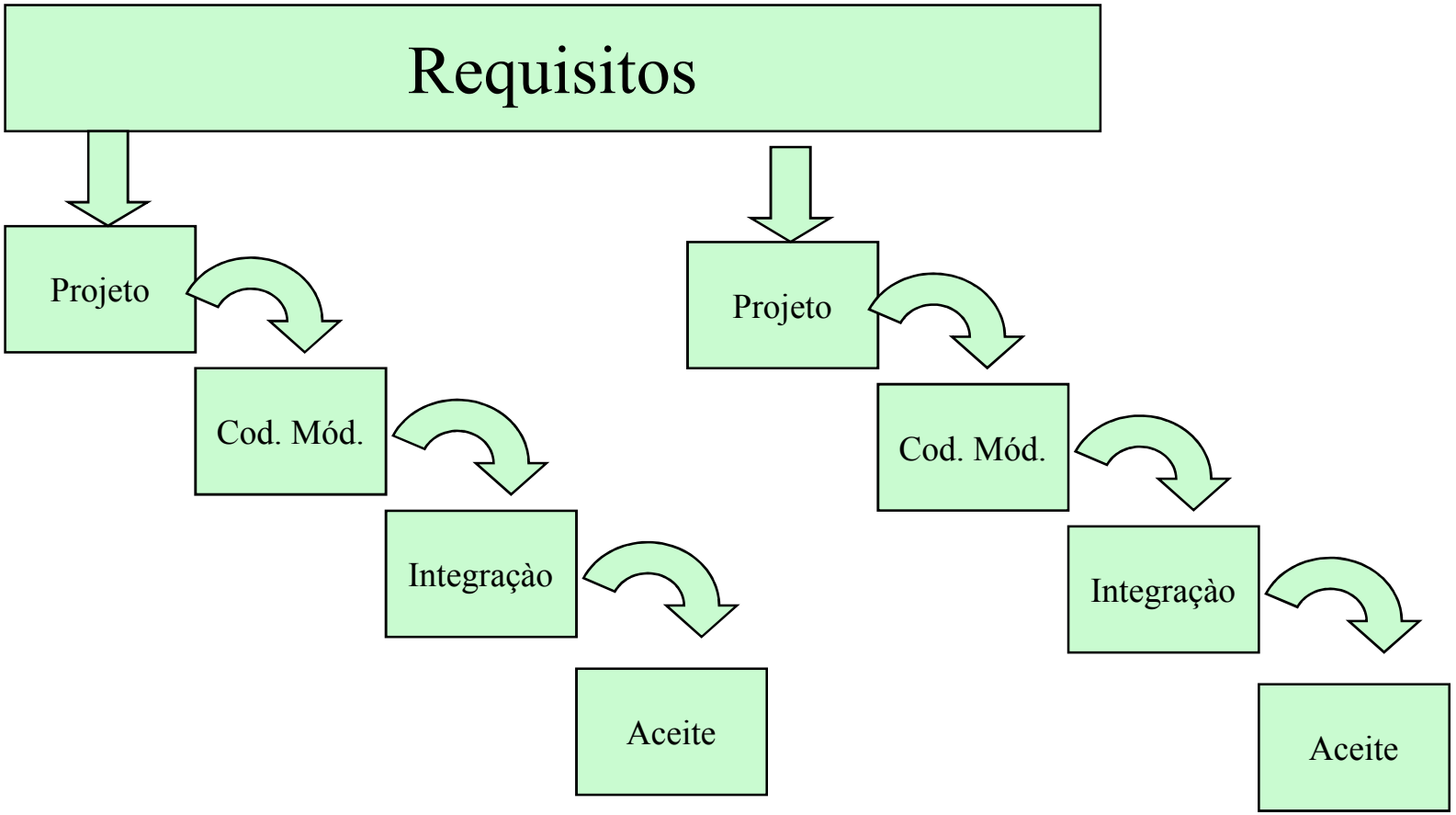
Modelo Cascata: Desenvolver é...

- Tentar um pouco de várias alternativas;
- Desenvolver a avaliar projetos;
- Avaliar a viabilidade dos requisitos, contrastando diversos projetos, aprendendo com as falhas para chegar a uma solução satisfatória.

Modelo Cascata: Resumo

- Clássico → Mais antigo;
- Sequencial;
- Gerenciamento Simples;
- Incompatível com Realidade Atual;
- Problema → Requisitos (uma única etapa);
- Retornos ???;
- Testes (final do processo);
- Erros Sutis – atraso no cronograma.

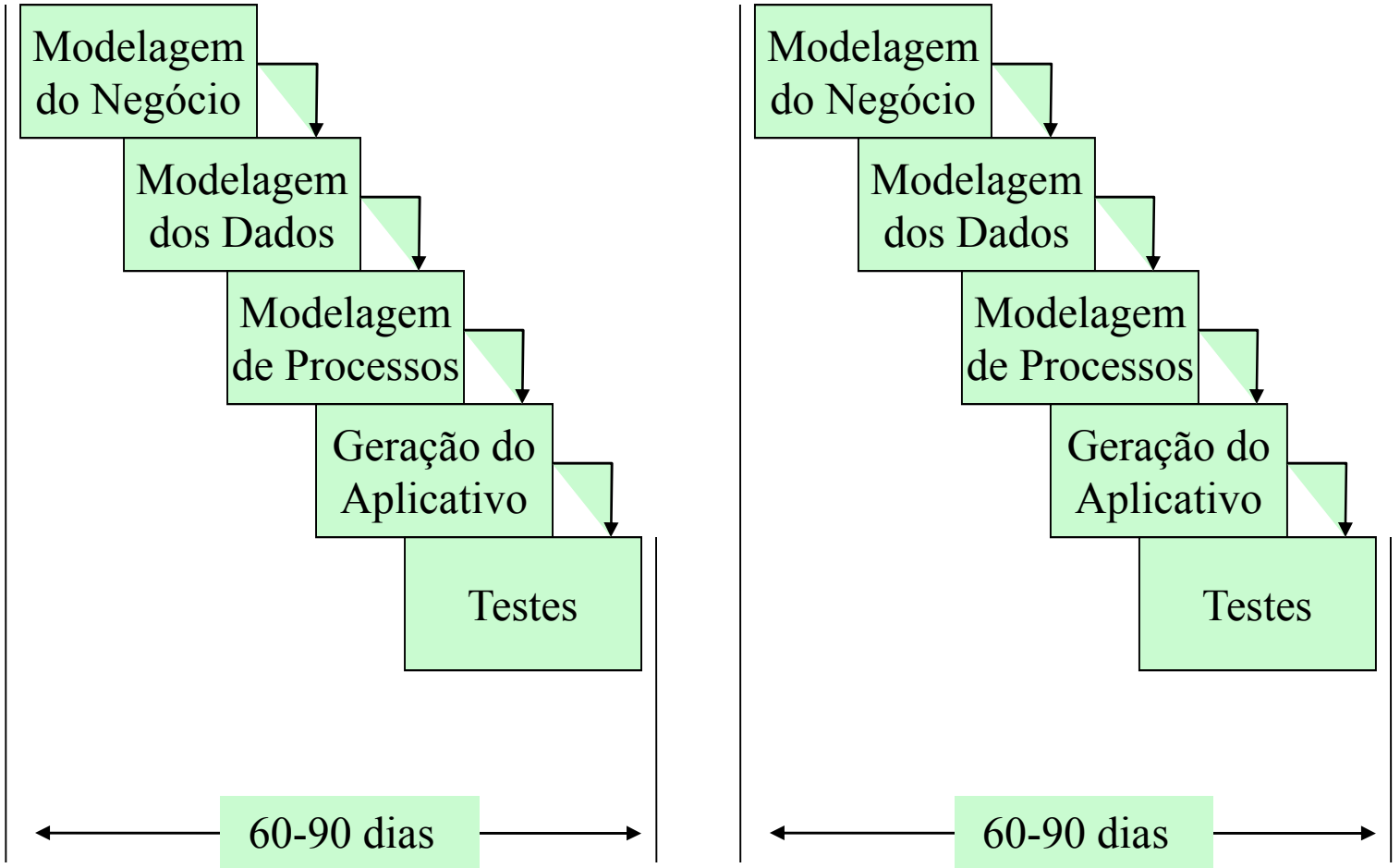
Modelo Incremental



Modelo Incremental

- Variante do Modelo Cascata
- Decomposição da Fase de Projeto
- Atividades em Paralelo
- Desenvolvimento em fases
- Entrega do sistema em partes
- Permissão para que o usuário utilize alguns recursos enquanto outros estão sendo desenvolvidos

Modelo Incremental RAD
(Rapid Application Development)



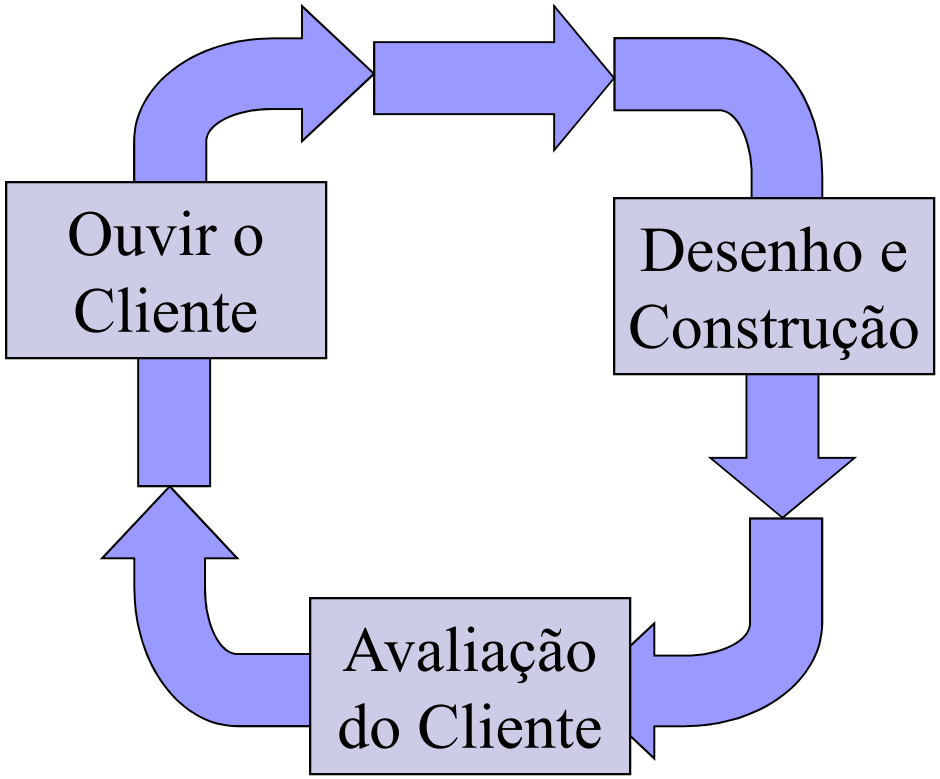
Modelo Incremental RAD

1. Modelagem do negócio: definição das atividades a serem executadas e seus requisitos de informação
2. Modelagem dos dados – Definição dos objetos de dados que suportam o negócio
3. Modelagem do tratamento da informação – Descrição dos processos de manipulação dos objetos de dados
4. Geração da aplicação – usando técnicas de geração de código e bibliotecas de componentes
5. Testes – Tempo de testes reduzido devido ao uso de componentes

Modelo Incremental RAD

- Pontos positivos
  - uso de componentes
  - redução do tempo
- Pontos negativos
  - tamanho da equipe
  - necessidade de comprometimento
  - não adequada a projetos de risco

Modelo por Prototipação



Prototipação é uma boa opção para a definição de requisitos

Modelo por Prototipação

- Pode ser usado tanto como um modelo específico a ser seguido (base de um processo efetivo), quanto como um sub-processo de outro modelo;
- Construído rapidamente para que questões sejam entendidas ou esclarecidas;
- Possibilita examinar antecipadamente os requisitos.

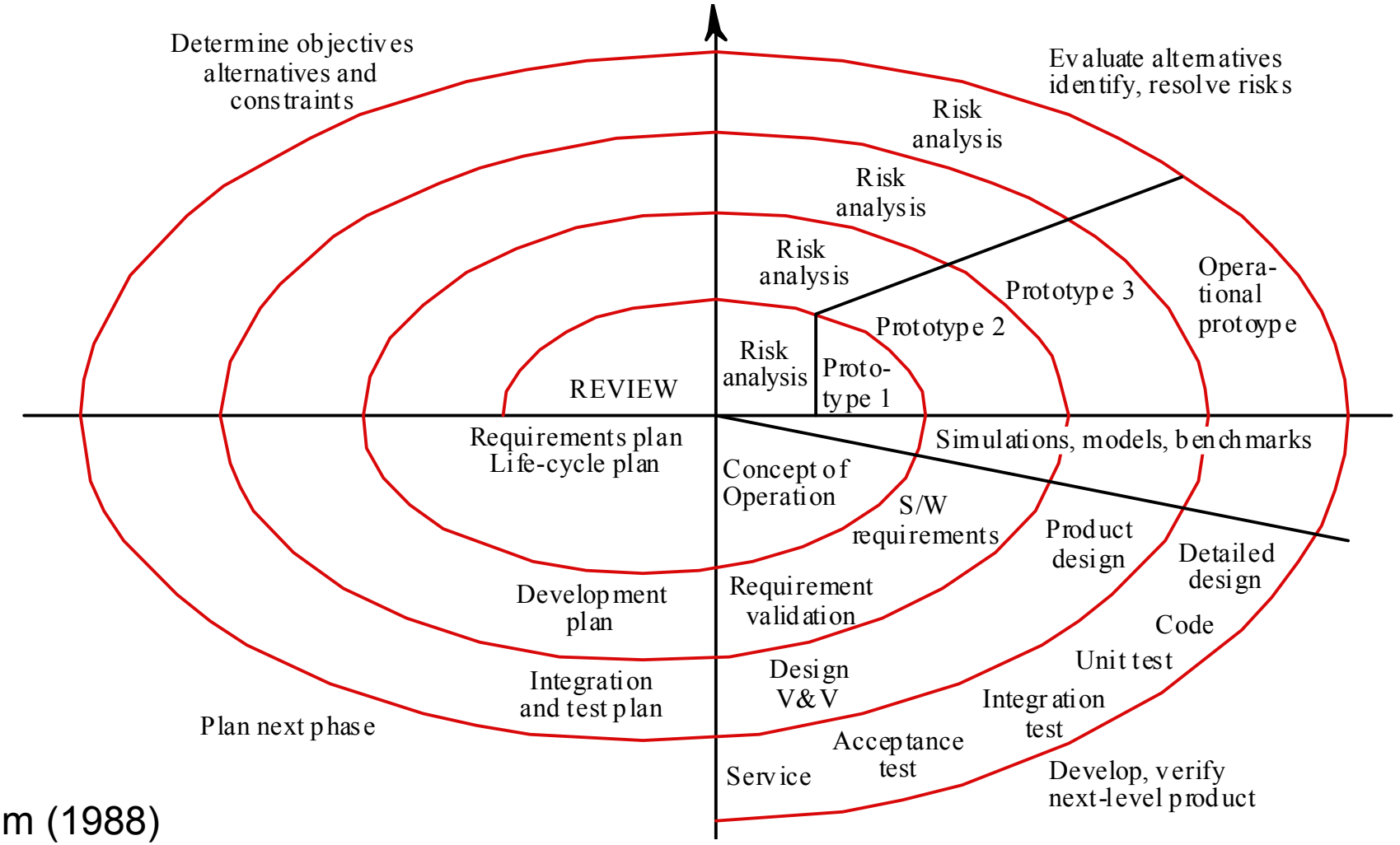
Modelo por Prototipação

- Reduz os riscos e as incertezas do desenvolvimento. Exemplo:
  - começa com um conjunto simples de requisitos
  - os usuários fazem experimentações e decidem o que querem
  - os requisitos são revisados, alterados, detalhados, documentados e o sistema passa a ser codificado
  - novamente as alternativas são discutidas e assim por diante até a entrega do software

Modelo por Prototipação

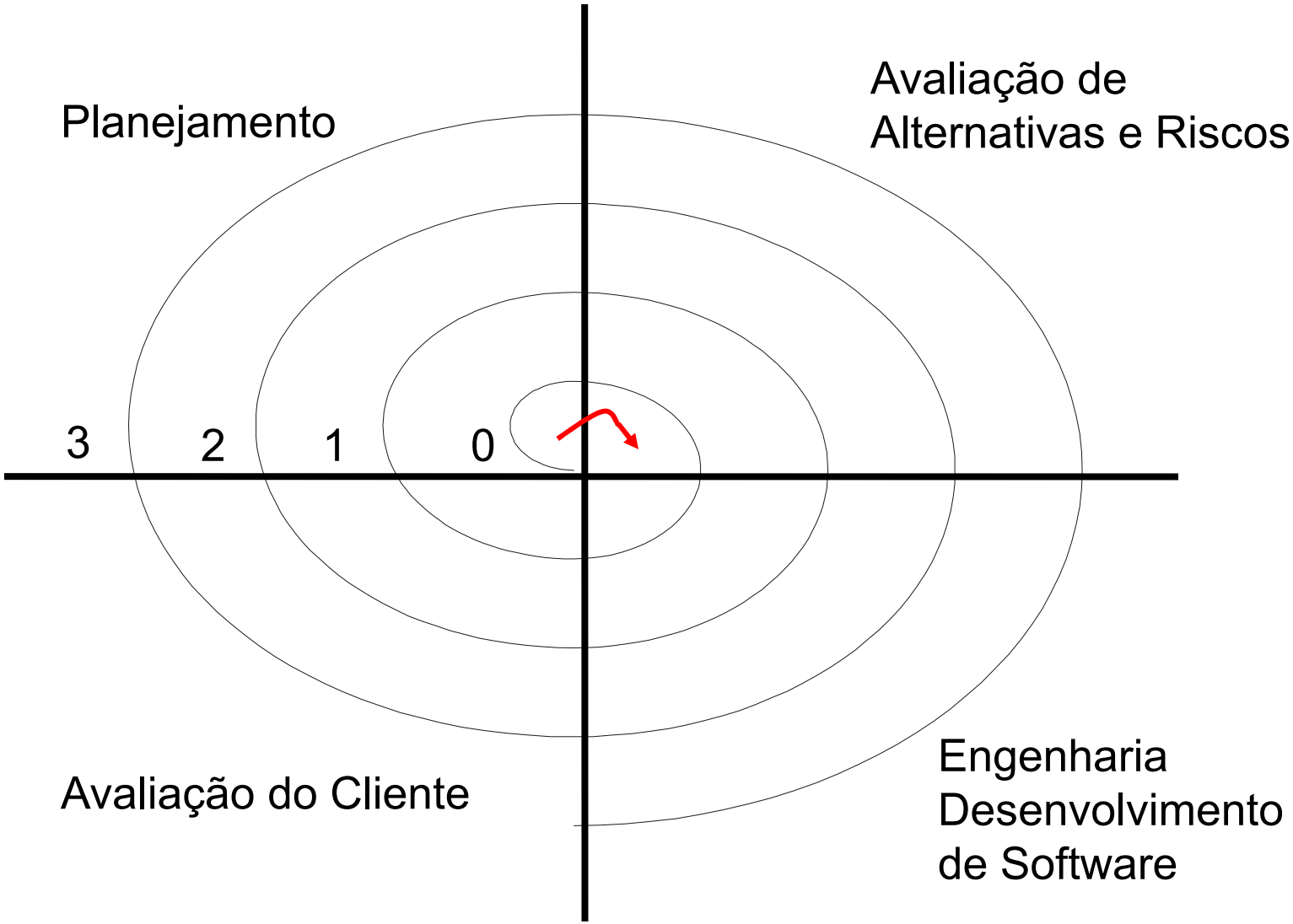
- Pontos positivos
  - grande interação com o usuário
  - qualidade da definição da interface
- Pontos negativos
  - expectativa do usuário
  - compromissos com a tecnologia

Modelo Espiral



Boehm (1988)

Modelo Espiral



Modelo Espiral

- Cada volta ao longo da espiral gera:
  - um protótipo
  - versão mais sofisticada;
- Permite ao desenvolvedor:
  - utilizar a prototipação em qualquer estágio de evolução do produto
  - manter a sistemática sugerida pelo ciclo de vida clássico.

Modelo Espiral

- Inicia com Requisitos e um Projeto inicial para o desenvolvimento (incluindo um orçamento, restrições e alternativas para o pessoal, o projeto e o ambiente de desenvolvimento);
- Combina as atividades de desenvolvimento com gerenciamento de risco;

Modelo Espiral

- Avaliar riscos e criação de protótipos alternativos antes de ser produzido um documento de “concepção das operações”;
- O objetivo é descrever em alto nível como o sistema deverá funcionar.

Modelo Espiral

- Especificação e Detalhamento de Requisitos (o completo quanto possível)
  - 1ª iteração – o produto é a concepção das operações;
  - 2ª iteração - os requisitos;
  - 3ª iteração - o desenvolvimento do sistema produz o projeto;
  - 4ª iteração - habilita os testes.

Modelo Espiral

- Em cada iteração, a análise de riscos pondera diferentes alternativas em face dos requisitos e das restrições.
- A prototipação verifica a viabilidade e a adequação, antes que haja a decisão por alguma alternativa.
- Quando riscos são identificados, o gerenciamento do projeto decide como eliminar ou minimizar cada risco.

Referências

- PRESSMAN, Roger S.; **Engenharia de Software**. São Paulo: Pearson, 2006.
- SOMMERVILLE, Ian; **Engenharia de Software**. São Paulo: Pearson, 2004.
- Macoratti. **O processo de Software.** <http://www.macoratti.net/proc_sw1.htm>