Fundamentos de Engenharia de Software

CICLO DE VIDA

Prof. Sergio Akio Tanaka

Ciclo de Vida

- A Engenharia de Software se preocupa com o software como produto. Como todo produto industrial, o software tem um ciclo de vida:
  - Ele é concebido a partir da percepção de uma necessidade;
  - É desenvolvido, transformando-se em um conjunto de itens entregue a um cliente;
  - Entra em operação, sendo usado dentro de algum processo de negócio, e sujeito a atividades de manutenção, quando necessário;
  - É retirado de operação, ao final de sua vida útil.

Ciclo de Vida - Conceitos

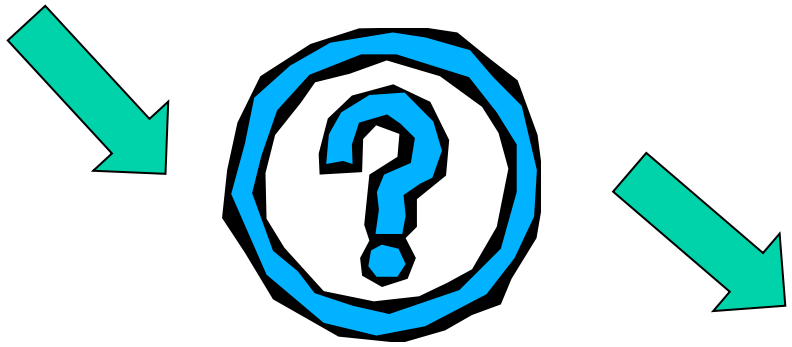
- É Considerado o intervalo de tempo decorrido desde o momento da concepção de um software até sua obsolescência;
- É identificado como um conjunto de fases ou etapas que representam uma evolução desde o nascimento da necessidade de criação de um software até a sua descontinuidade ou morte.

Objetivo

- Sugerir uma ordenação das atividades existentes no desenvolvimento e manutenção de software;
- Agregar qualidade: Boa qualidade no processo de desenvolvimento e no produto final de um software está relacionado com a escolha do modelo de ciclo de vida a ser adotado. Ver norma ISO 9000-3.

O Modelo “Codifica-Remenda”

Especificação
(???)



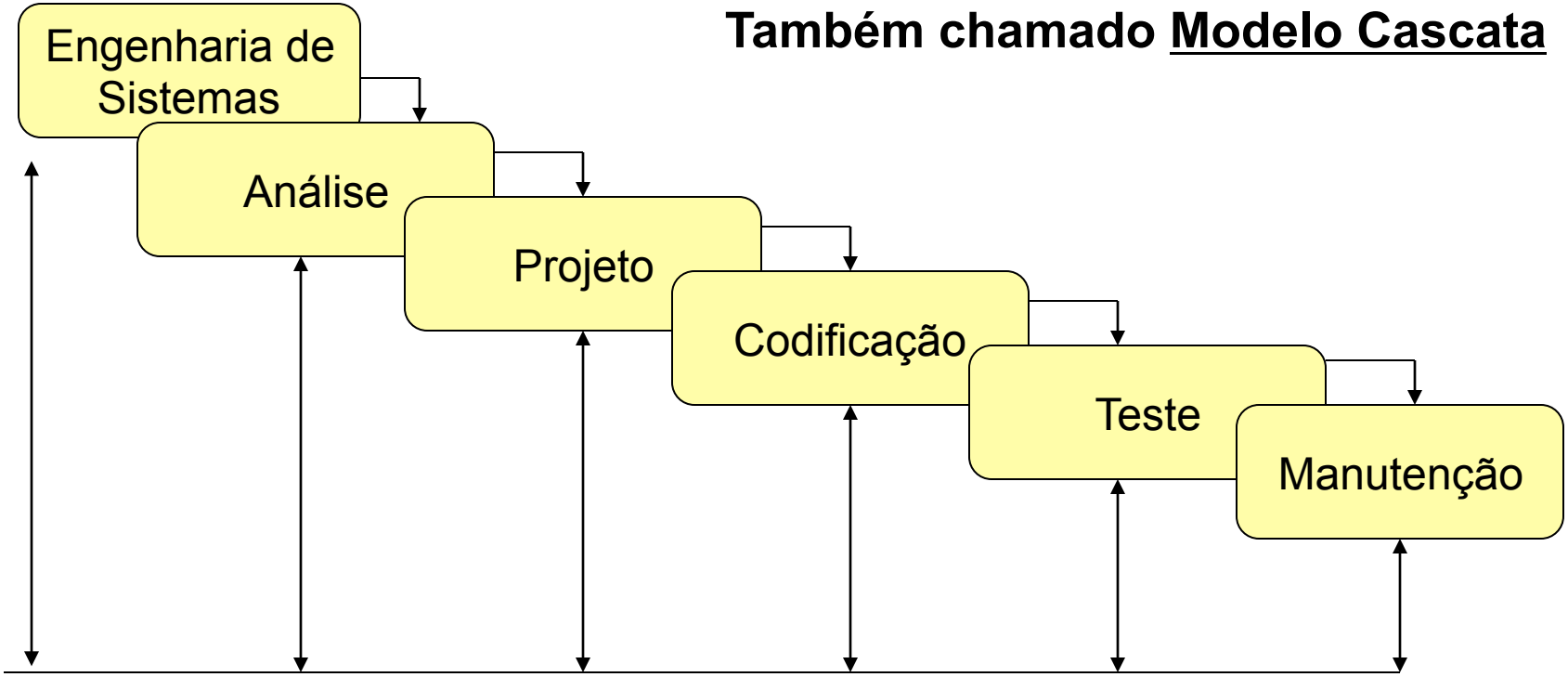
Produto

O Modelo "Codifica-Remenda"

- É o ciclo de vida mais caótico;
- Partindo apenas de uma especificação (ou nem isso), os desenvolvedores começam imediatamente a codificar, remendando à medida que os erros vão sendo descobertos;
- Infelizmente, é provavelmente o ciclo de vida mais usado;
- Para alguns desenvolvedores, esse modelo é atraente porque não exige nenhuma sofisticação técnica ou gerencial;
- Por outro lado, é um modelo de alto risco, impossível de gerir e que não permite assumir compromissos confiáveis.

O Ciclo de Vida Clássico

Também chamado <u>Modelo Cascata</u>



O Ciclo de Vida Clássico

Análise e engenharia de sistemas
(Análise de requisitos de software)

- Coleta dos requisitos em nível do sistema
- Identificação dos serviços e metas a serem atingidos
- Identificada a qualidade desejada para o sistema em termos de funcionalidade, desempenho, facilidade de uso, portabilidade, ...
- Preocupa-se em identificar quais requisitos sem se preocupar como eles serão implementados.

O Ciclo de Vida Clássico

Projeto

• Representação das funções do sistema em uma forma que possa ser transformada em um ou mais programas executáveis.

• Define-se: estrutura de dados, arquitetura de software, detalhes procedimentais e caracterização de interface.

Codificação

•O Projeto deve ser traduzido numa forma legível por máquina. Se o projeto for executado detalhadamente, a codificação pode ser executada mecanicamente

O Ciclo de Vida Clássico

Teste

- Concentra-se nos aspectos **lógicos internos** do software (todas as instruções testadas), e nos **funcionais externos** (testes para descoberta de erros).
- Pode incluir inspeção de código, para checagem de padrões de codificação, estilo de programação, verificação de desempenho.
- Testes de unidades (módulos) e de integração.

Manutenção

- O software sofrerá mudanças depois que for entregue ao cliente. Ocorrerão mudanças porque erros foram encontrados, porque o software deve ser adaptado a fim de acomodar mudanças em seu ambiente externo, ou porque o cliente exige acréscimos funcionais ou de desempenho.

O Ciclo de Vida Clássico

- Encoraja a especificação do sistema de início;
- Capacita o gerente a caminhar progressivamente e descobrir possíveis erros;
- Os produtos das fases anteriores são utilizados como base para as outras fases;
- Demanda a produção de uma série de documentos no decorrer do processo;
- Não indicado para sistemas com grande interação com o usuário;
- O modelo cascata é de baixa visibilidade para o cliente, que só recebe o resultado final do projeto.

O Modelo Prototipação Rápida Descartável

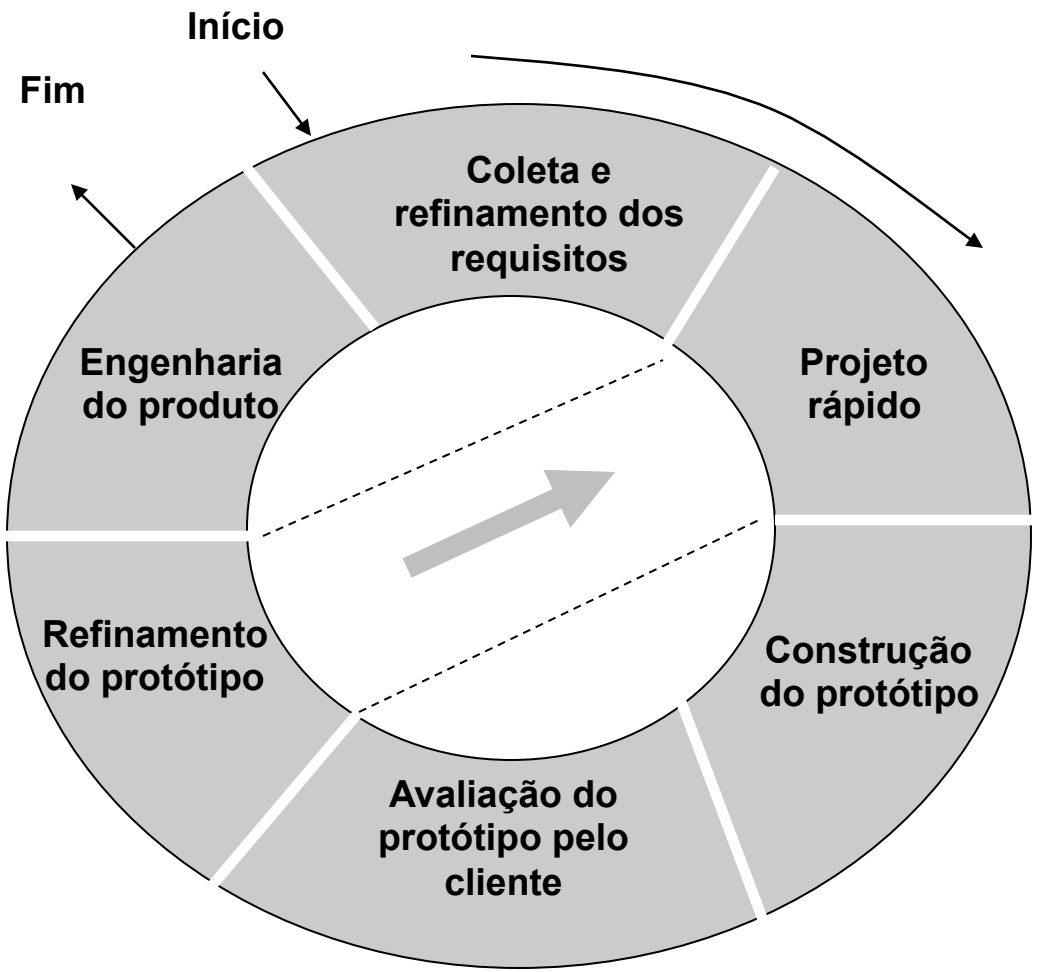
- Fornecer rapidamente uma versão para ser utilizada e avaliada pelo usuário;
- Não necessidade de se satisfazer todos os requisitos do produto final desde o início;
- Facilitar o desenvolvimento de produtos onde não se conhece totalmente o problema;
- O usuário não consegue especificar de uma forma clara os requisitos do sistema;
- Pode ser usado quando o sistema possuir muita interação com o usuário e se deseja avaliar a interface.

Prototipação

• O desenvolvedor pode estar incerto sobre a eficiência de um algoritmo, adaptação de um sistema operacional ou sobre a forma da interação homem-máquina;

O protótipo se concentra em fazer experimentos com os requisitos do usuário que não estão bem entendidos e envolve projeto, implementação e teste, mas não de maneira formal ou completa.

Prototipação



O Modelo Incremental

- Os requisitos são conhecidos;
- Não se quer implementar todo o sistema de uma vez (sistema dividido em subsistemas ou módulos);
- Escolhem-se prioridades de implementação. Fatores como tempo e disponibilidade de recursos e pessoal podem influenciar nas prioridades;
- Deseja ter rapidamente uma versão operacional mesmo que não tenha todas as funções.
- A cada ciclo uma nova versão operacional do sistema será produzida

O Modelo Incremental

- Como as funções prioritárias são entregues primeiro e os incrementos são integrados a elas, e inevitável que as funções de sistemas mais importantes passem pela maior parte dos testes. Isto significa que e menos provável que os clientes encontrem falhas de software na parte mais importante do sistema.

O Modelo Espiral

Também conhecido como Paradigma de Boehm.

• Engloba as melhores características do ciclo de vida clássico e da prototipação, adicionando um novo elemento — a análise de risco — que não existe nos outros paradigmas.

Riscos são circunstâncias adversas que podem atrapalhar o processo de desenvolvimento e a qualidade do produto a ser desenvolvido.

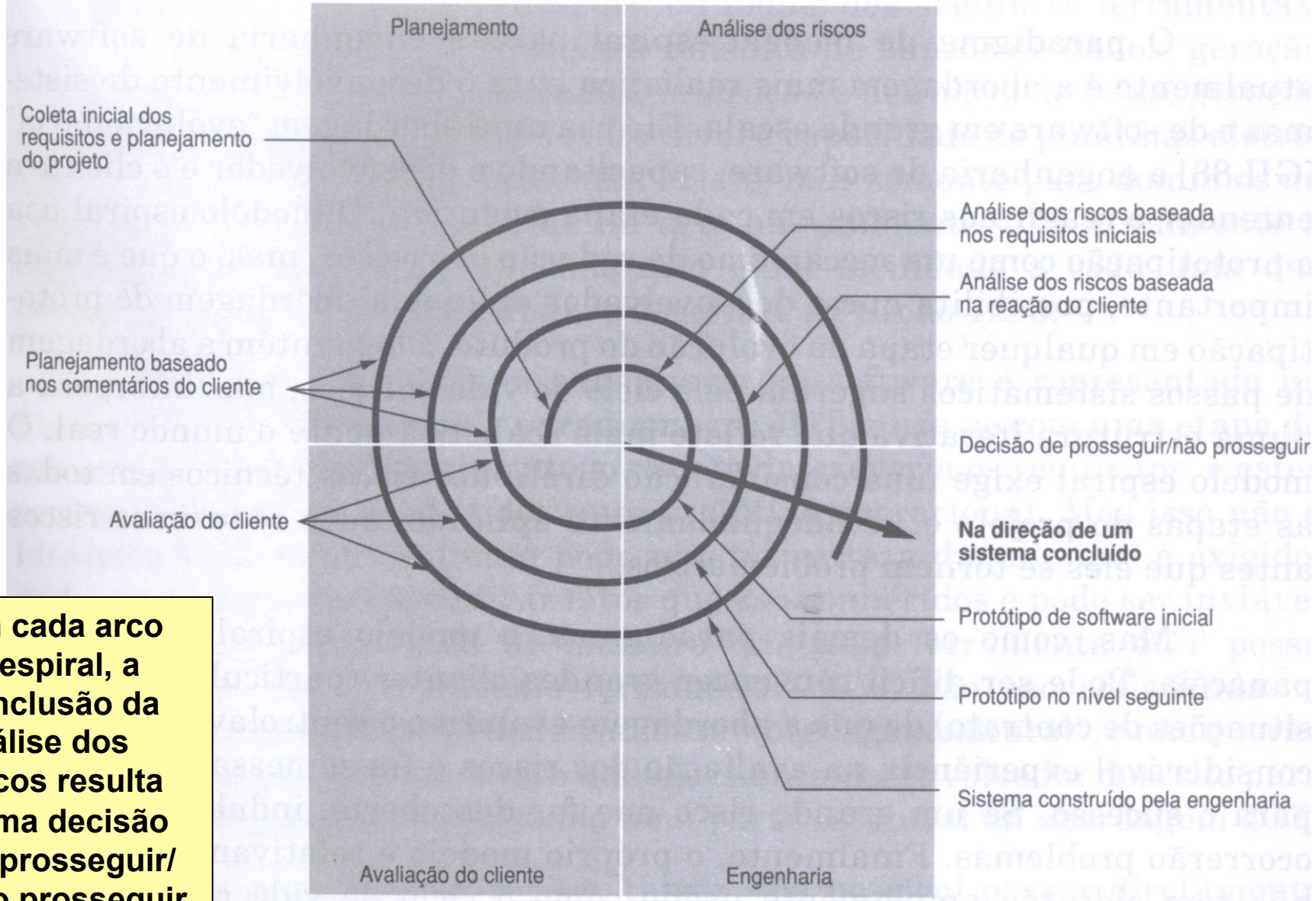
O paradigma, representado pela espiral, define quatro atividades principais representadas pelos quadrantes:
Planejamento, Análise dos riscos, Engenharia e Avaliação do Cliente

Versões progressivamente mais completas do software são construídas.

O Modelo Espiral

- O produto e desenvolvido em uma serie de iterações;
- Cada nova iteração corresponde a uma volta na espiral;
- Isso permite construir produtos em prazos curtos, com novas características e **recursos** que são **agregados** na medida em que a experiência descobre suas necessidades;
- As atividades de manutenção são usadas para ***identificar problemas***; seus registros fornecem dados para definir os **requisitos das próximas liberações;**
- O principal problema do ciclo de vida em espiral é que ele **requer gestão muito sofisticada** para ser previsível e confiável.

O Modelo Espiral



Em cada arco da espiral, a conclusão da análise dos riscos resulta numa decisão de prosseguir/ não prosseguir.

Modelo Prototipagem Evolutiva

- É uma variante do modelo em espiral;
- A espiral é usada não para desenvolver o produto completo, mas para **construir uma série de versões provisórias** que são chamadas de protótipos ;
- Em vez de ter as atividades de **especificação , desenvolvimento e validação** em separado , todo esse trabalho e realizado concorrentemente com um rápido feedback por meio dessas atividades;

Modelo Prototipagem Evolutiva

- Os protótipos cobrem cada vez mais requisitos, até que se atinja o produto desejado;
- A prototipagem evolutiva permite que os requisitos sejam definidos progressivamente, e apresenta alta flexibilidade e visibilidade para os clientes
- Requer gestão sofisticada e o desenho deve ser de excelente qualidade, para que a estrutura do produto não se degenere ao longo dos protótipos.