AULA 9:

MODELOS DE PROCESSO DE SOFTWARE – PARTE II

Modelos de Processo de Software

- Modelo Sequencial Linear
  - Também chamado Ciclo de Vida Clássico ou Modelo Cascata.
- O Paradigma de Prototipação
- Técnicas de Quarta Geração
- Modelo RAD (Rapid Application Development)
- Modelos de Métodos Formais
- Modelos Evolutivos de Processo de Software
  - Modelo Incremental.
  - Modelo Espiral.
  - Modelo de Montagem de Componentes.
  - Modelo de Desenvolvimento Concorrente.

Modelos de Processo de Software

- **Modelo Sequencial Linear**
  - **Também chamado Ciclo de Vida Clássico ou Modelo Cascata.**
- **O Paradigma de Prototipação**
- Técnicas de Quarta Geração
- **Modelo RAD (Rapid Application Development)**
- Modelos de Métodos Formais
- Modelos Evolutivos de Processo de Software
  - Modelo Incremental.
  - Modelo Espiral.
  - Modelo de Montagem de Componentes.
  - Modelo de Desenvolvimento Concorrente.

Modelos Evolutivos de Processo de Software

- Existem situações em que a engenharia de software necessita de um modelo de processo que possa acomodar um produto que evolui com o tempo.
- Indicado utilizar quando os requisitos de produto e de negócio mudam conforme o desenvolvimento.
  - Quando uma data de entrega apertada (mercado) - impossível a conclusão de um produto completo.
  - Quando um conjunto de requisitos importantes é bem conhecido, porém os detalhes ainda devem ser definidos.
- Modelos evolutivos são iterativos.
- Possibilitam o desenvolvimento de versões cada vez mais completas do software.

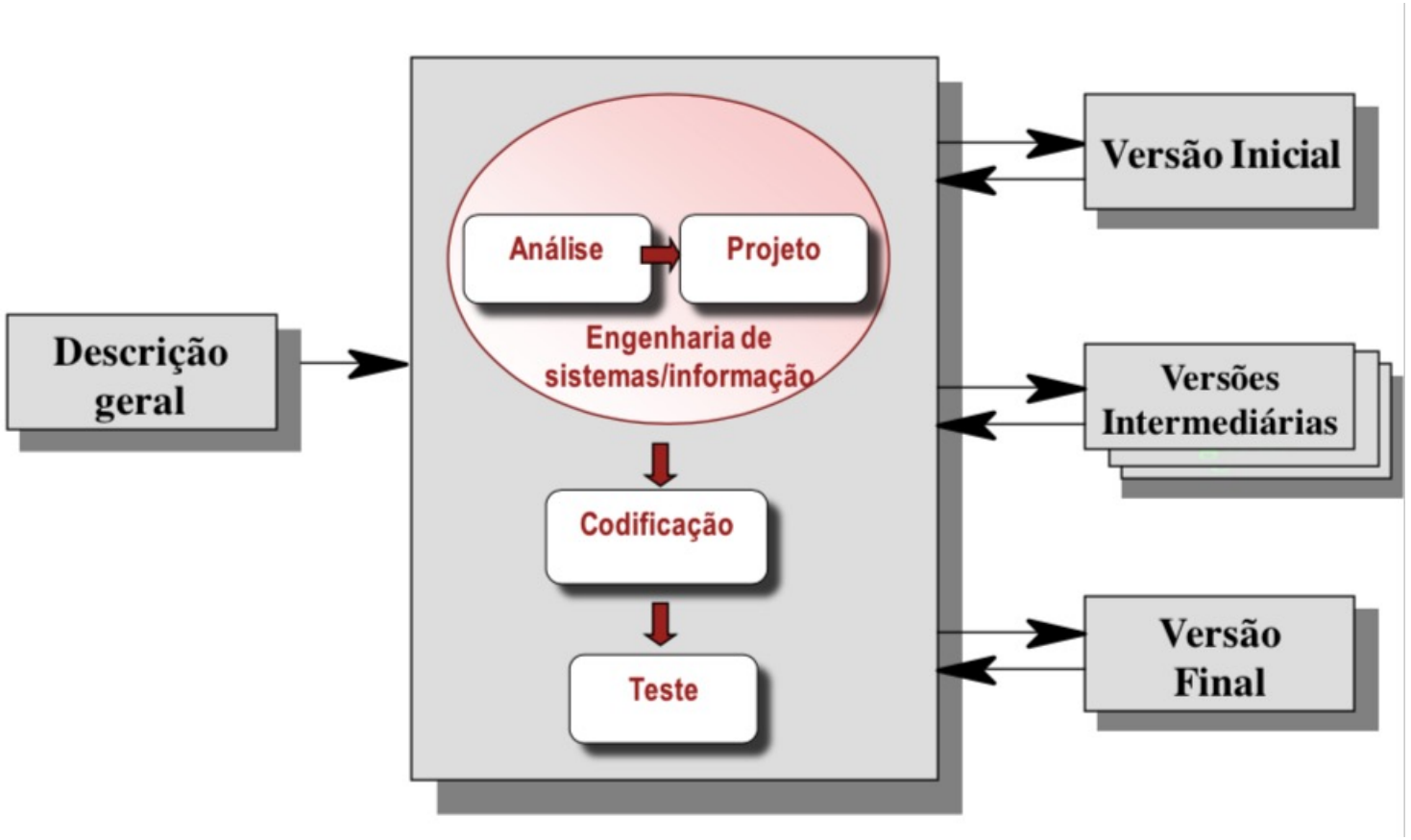
Modelos de Processo de Software

- Modelo Sequencial Linear
  - Também chamado Ciclo de Vida Clássico ou Modelo Cascata.
- O Paradigma de Prototipação
- Técnicas de Quarta Geração
- Modelo RAD (Rapid Application Development)
- Modelos de Métodos Formais
- Modelos Evolutivos de Processo de Software
  - Modelo Incremental.
  - Modelo Espiral.
  - Modelo de Montagem de Componentes.
  - Modelo de Desenvolvimento Concorrente.

O Modelo Incremental

- O modelo incremental combina elementos do modelo cascata (aplicado repetidamente) com a filosofia iterativa da prototipação.
- O objetivo é trabalhar junto do usuário para descobrir seus requisitos, de maneira incremental, até que o produto final seja obtido.

O Modelo Incremental



O Modelo Incremental

- A **versão inicial** é frequentemente o núcleo do produto (a parte mais importante).
- A evolução acontece quando novas características são adicionadas à medida que são sugeridas pelo usuário.
- Este modelo é importante quando é difícil estabelecer a priori uma especificação detalhada dos requisitos.
- O modelo incremental é mais apropriado para sistemas pequenos.
- As novas versões podem ser planejadas de modo que os riscos técnicos possam ser administrados.
- Exemplo: disponibilidade de determinado hardware.

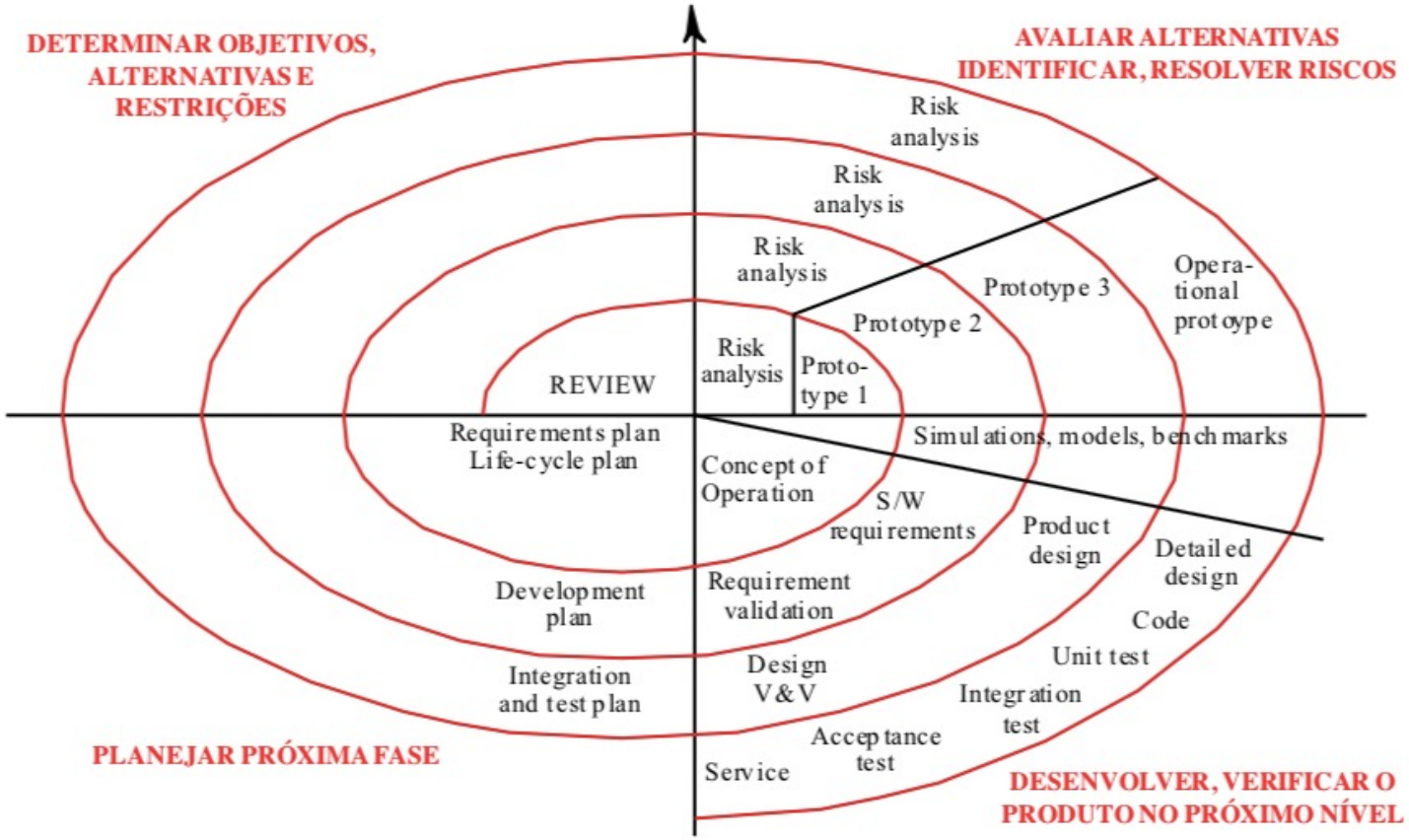
Modelos de Processo de Software

- Modelo Sequencial Linear
  - Também chamado Ciclo de Vida Clássico ou Modelo Cascata.
- O Paradigma de Prototipação
- Técnicas de Quarta Geração
- Modelo RAD (Rapid Application Development)
- Modelos de Métodos Formais
- Modelos Evolutivos de Processo de Software
  - Modelo Incremental.
  - Modelo Espiral.
  - Modelo de Montagem de Componentes.
  - Modelo de Desenvolvimento Concorrente.

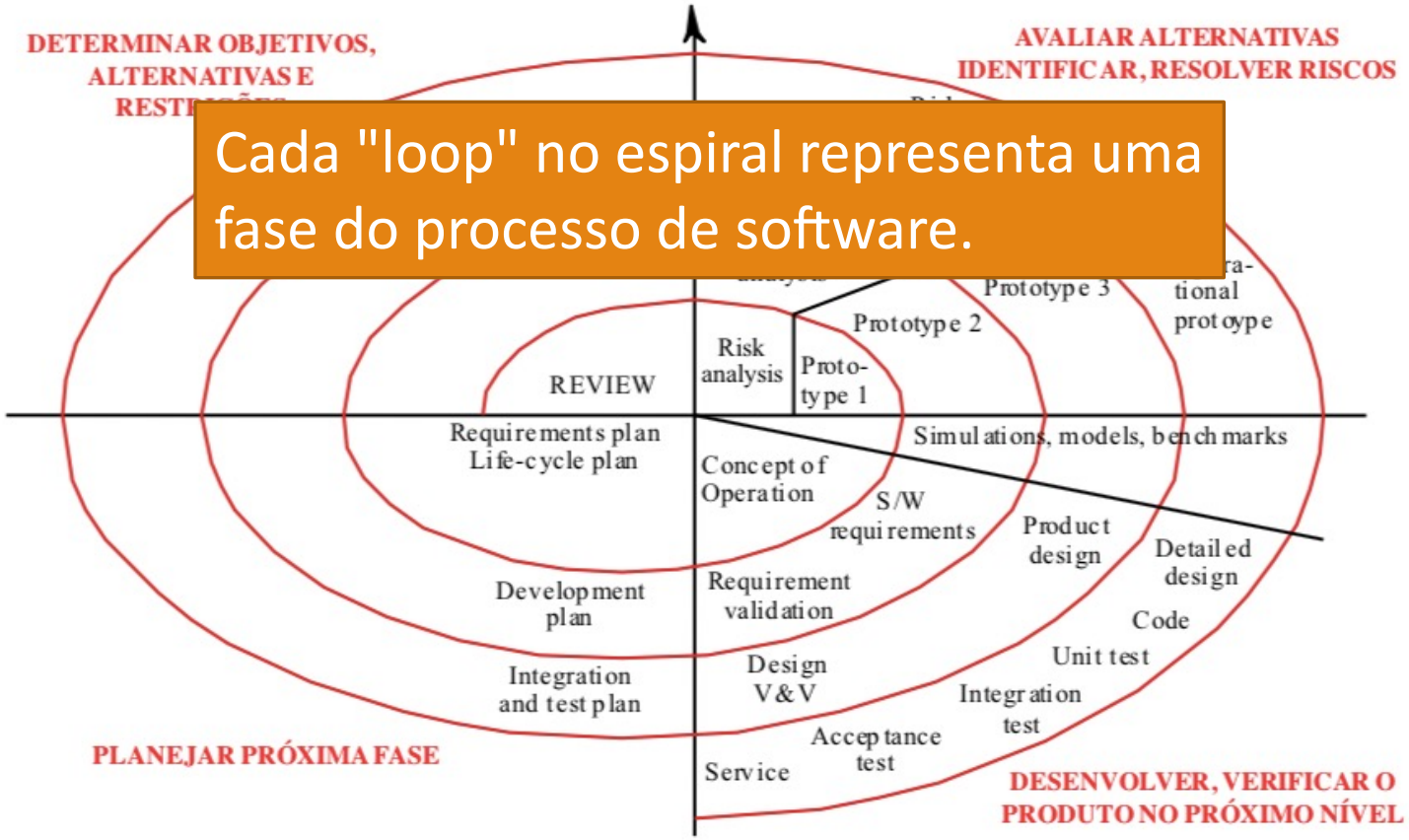
O Modelo Espiral

- O modelo espiral acopla a natureza iterativa da prototipação com os aspectos controlados e sistemáticos do modelo cascata.
- O modelo espiral é dividido em uma série de atividades de trabalho ou regiões de tarefa.
- Existem tipicamente de 3 a 6 regiões de tarefa.

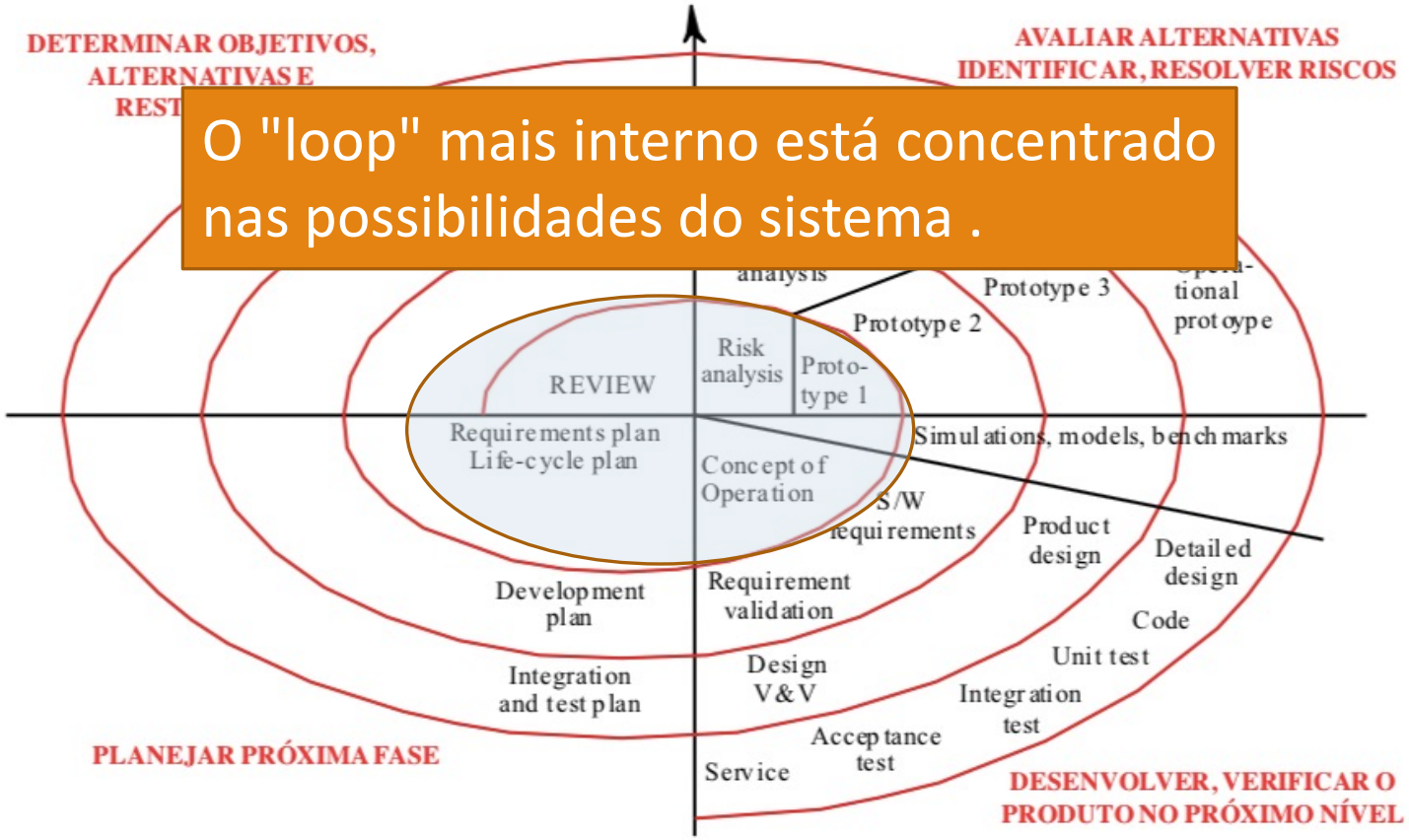
O Modelo Espiral (4 Regiões)



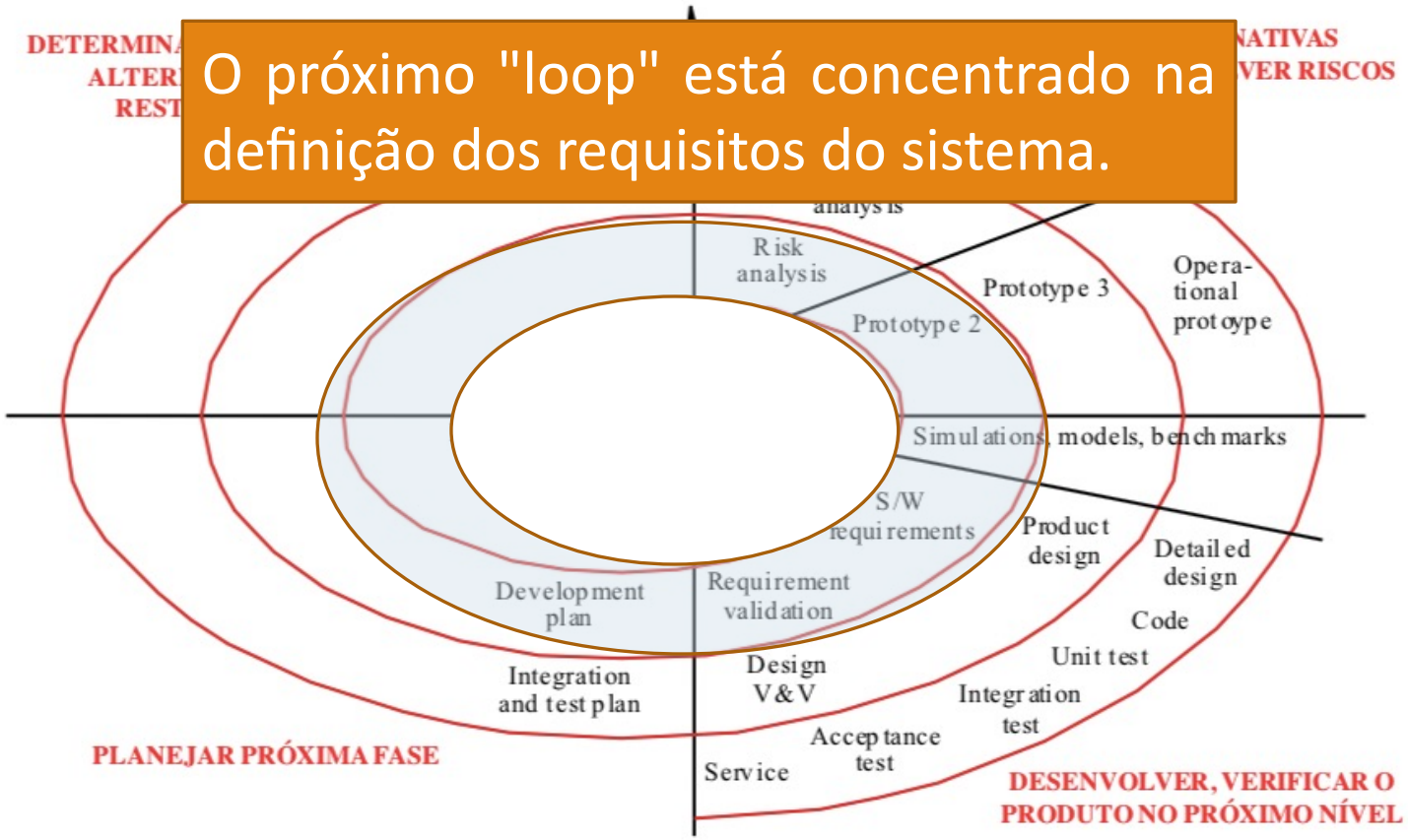
O Modelo Espiral (4 Regiões)



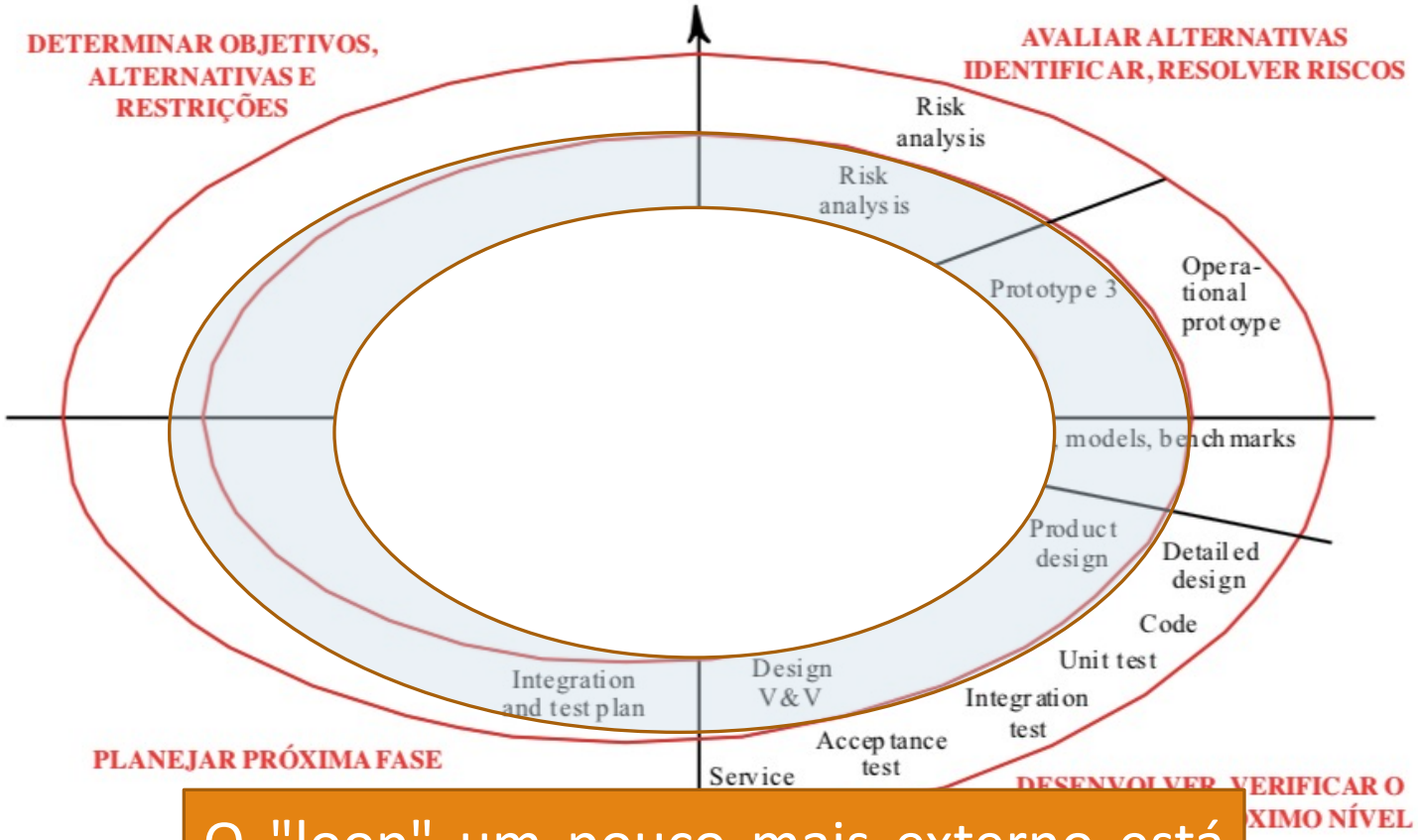
O Modelo Espiral (4 Regiões)



O Modelo Espiral (4 Regiões)

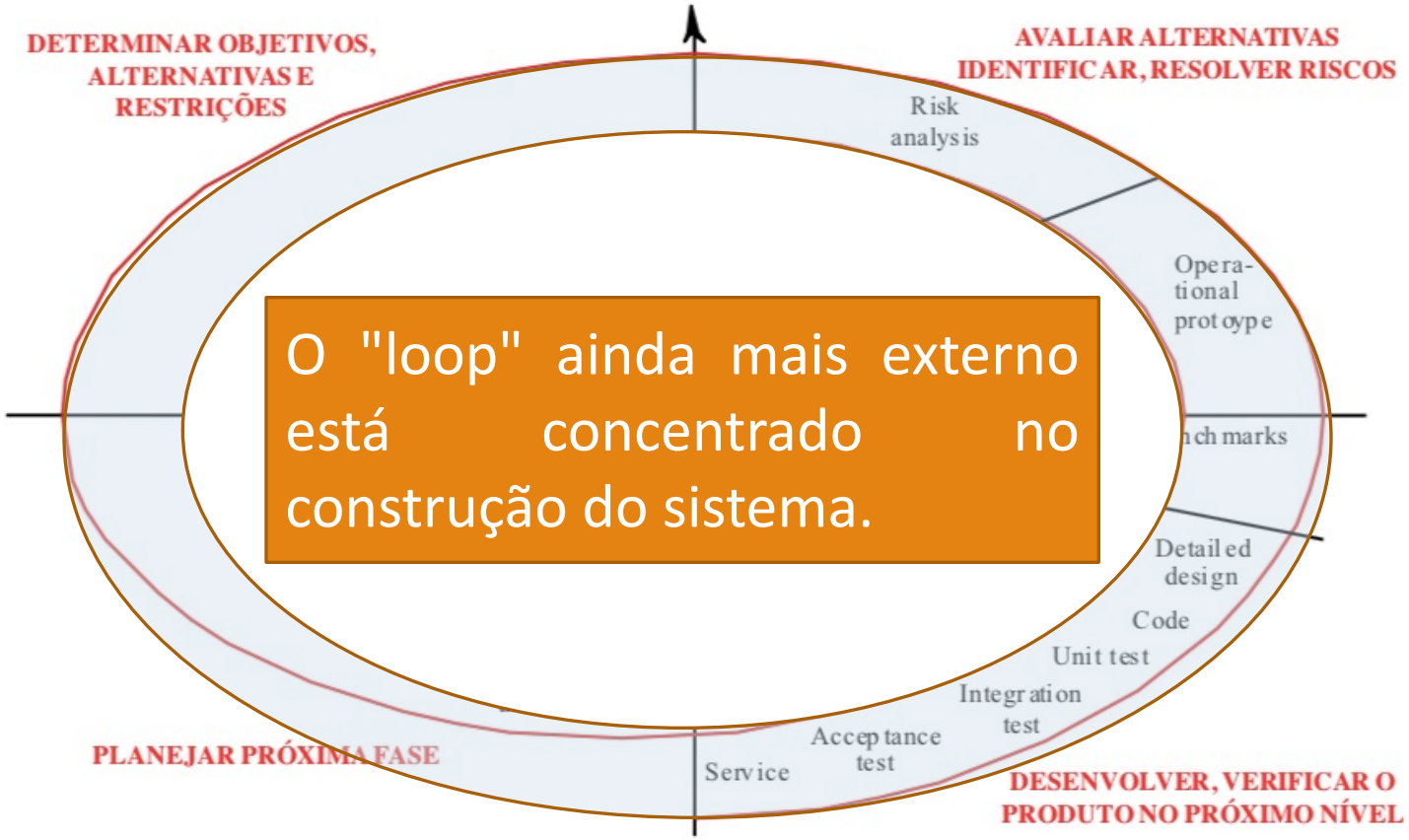


O Modelo Espiral (4 Regiões)



O "loop" um pouco mais externo está concentrado no projeto do sistema.

O Modelo Espiral (4 Regiões)



O "loop" ainda mais externo está concentrado no construção do sistema.

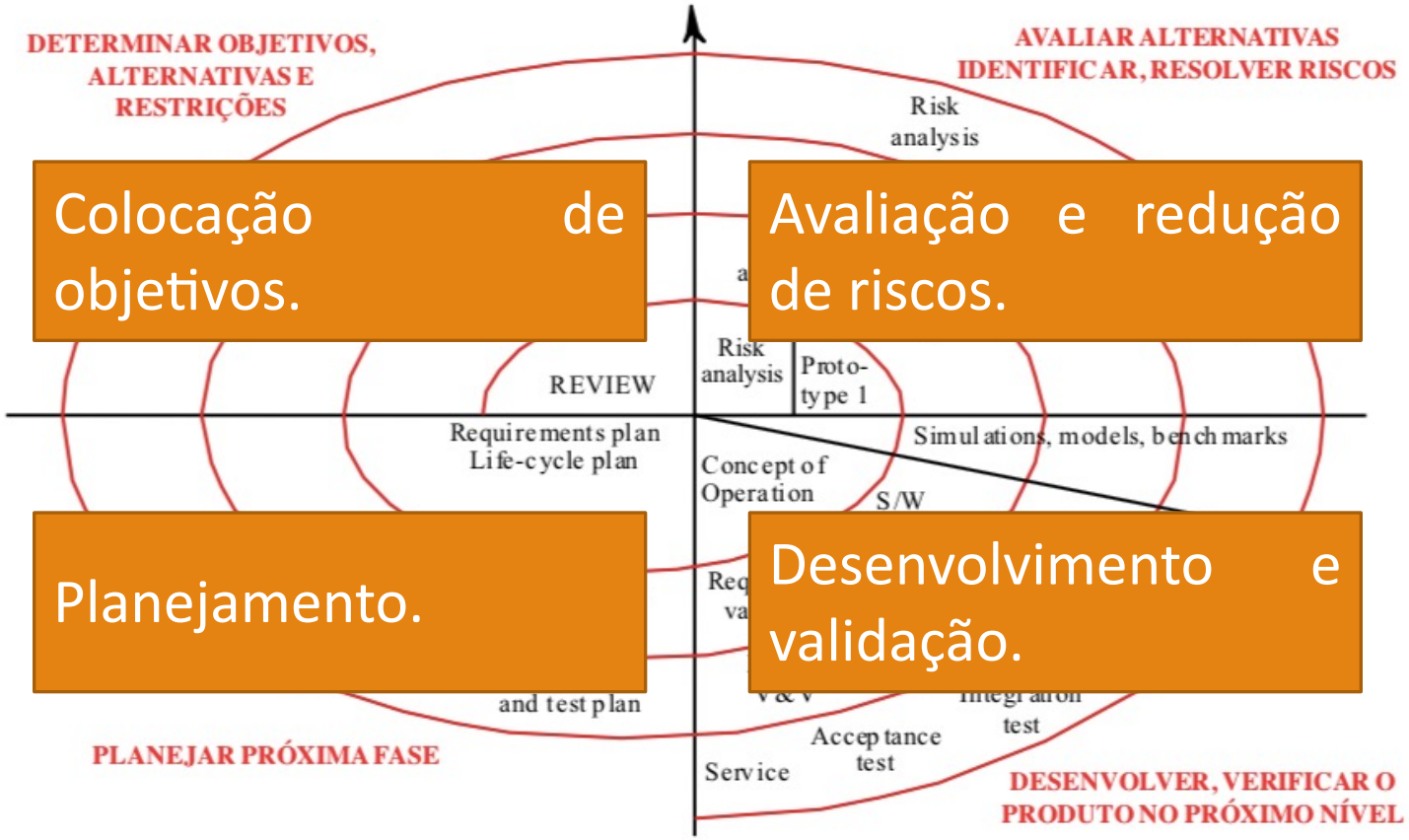
O Modelo Espiral

- Não existem fases fixas no modelo.
- As fases mostradas na figura são meramente exemplos.
- A gerência decide como estruturar o projeto em fases.

O Modelo Espiral (4 Regiões)

Cada "loop" do espiral é dividido em 4 setores.

O Modelo Espiral (4 Regiões)



O Modelo Espiral

Colocação de Objetivos

- São definidos objetivos específicos para a fase do projeto.
- São identificadas restrições sobre o processo e o produto.
- É projetado um plano de gerenciamento detalhado.
- São identificados riscos do projeto.
- Dependendo dos riscos, estratégias alternativas podem ser planejadas.

Planejamento

- O projeto é revisto e é tomada uma decisão de continuidade.
- Se é decidido continuar, são projetados planos para a próxima fase do projeto (próximo "loop").

O Modelo Espiral

Avaliação e redução de riscos

- Para cada um dos riscos identificados, uma análise detalhada é executada.
- Passos são tomados para reduzir o risco.

Desenvolvimento e Validação

- Depois da avaliação do risco, um modelo de desenvolvimento é escolhido para o sistema.

O Modelo Espiral

Resumo

- Engloba as melhores características do ciclo de vida Clássico e da Prototipação, adicionando um novo elemento: a Análise de Risco.
- Segue a abordagem de passos sistemáticos do Ciclo de Vida Clássico incorporando-os numa estrutura iterativa que reflete mais realisticamente o mundo real.
- Usa a Prototipação, em qualquer etapa da evolução do produto, como mecanismo de redução de riscos.

O Modelo Espiral

Comentários

- É atualmente, a abordagem mais realística para o desenvolvimento de software em grande escala.
- Usa uma abordagem que capacita o desenvolvedor e o cliente a entender e reagir aos riscos em cada etapa evolutiva.
- Pode ser difícil convencer os clientes que uma abordagem "evolutiva" é controlável.
- Exige considerável experiência na determinação de riscos e depende dessa experiência para ter sucesso.

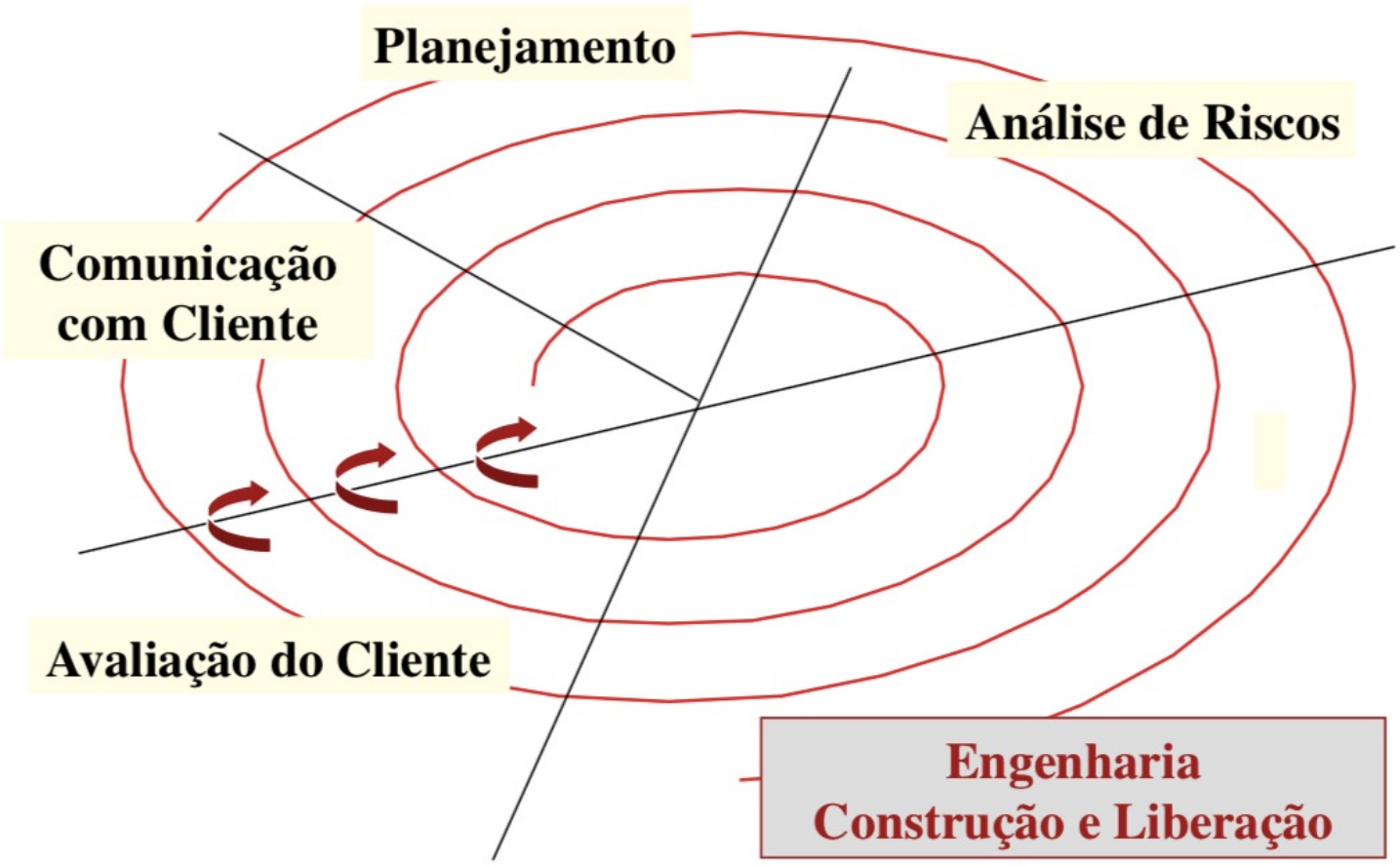
Modelos de Processo de Software

- Modelo Sequencial Linear
  - Também chamado Ciclo de Vida Clássico ou Modelo Cascata.
- O Paradigma de Prototipação
- Técnicas de Quarta Geração
- Modelo RAD (Rapid Application Development)
- Modelos de Métodos Formais
- Modelos Evolutivos de Processo de Software
  - Modelo Incremental.
  - Modelo Espiral.
  - Modelo de Montagem de Componentes.
  - Modelo de Desenvolvimento Concorrente.

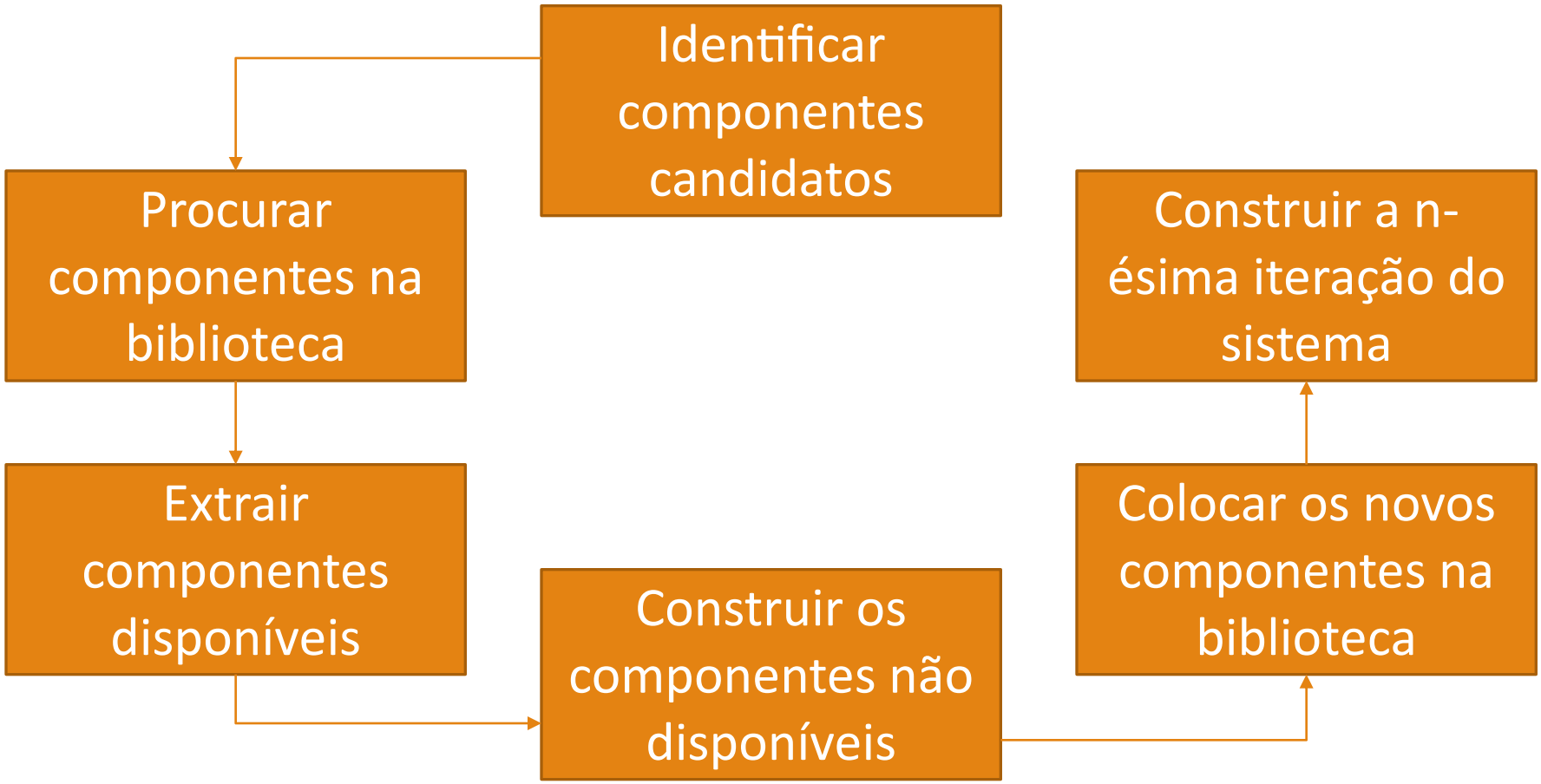
Modelo de Montagem de Componentes

- Utiliza tecnologias orientadas a objeto.
- Quando projetadas e implementadas apropriadamente, as classes orientadas a objeto são reutilizáveis em diferentes aplicações e arquiteturas de sistema.
- O modelo de montagem de componentes incorpora muitas das características do modelo espiral.

Modelo de Montagem de Componentes



Modelo de Montagem de Componentes



Modelo de Montagem de Componentes

- O modelo de montagem de componentes conduz ao **reuso** do software.
- A reusabilidade fornece uma série de benefícios:
  - Redução de 70% no tempo de desenvolvimento;
  - Redução de 84% no custo do projeto;
  - Índice de produtividade de 26.2 (normal da indústria é de 16.9).
- Esses resultados dependem da robustez da biblioteca de componentes.

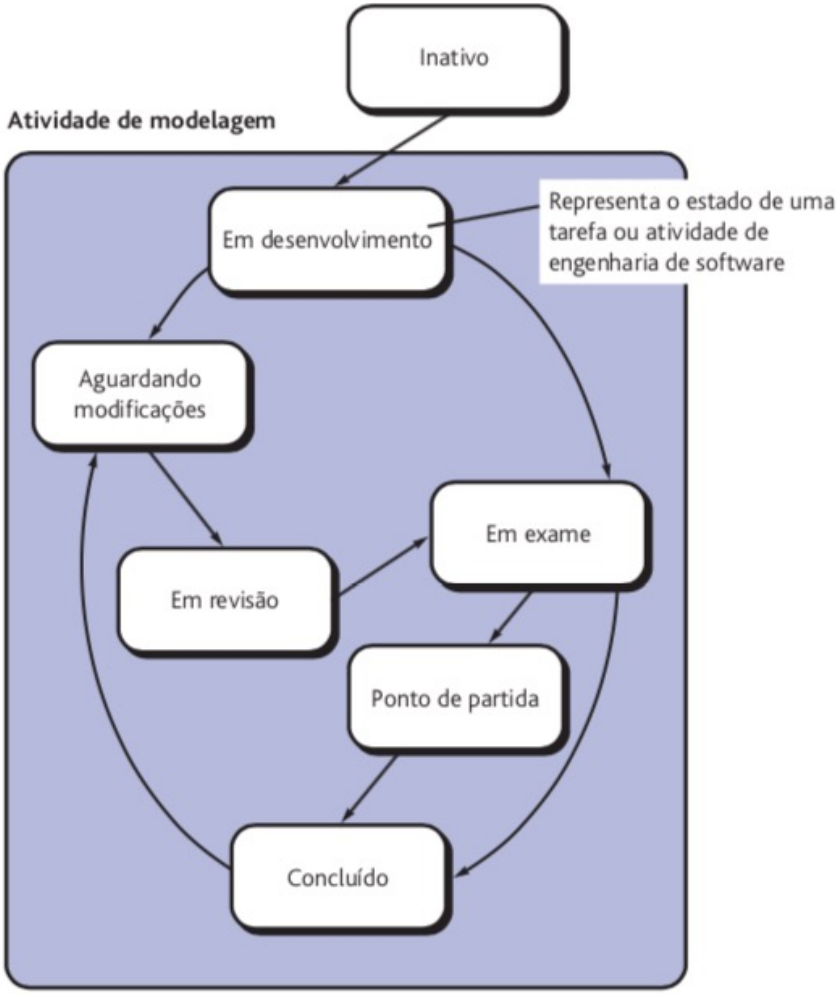
Modelos de Processo de Software

- Modelo Sequencial Linear
  - Também chamado Ciclo de Vida Clássico ou Modelo Cascata.
- O Paradigma de Prototipação
- Técnicas de Quarta Geração
- Modelo RAD (Rapid Application Development)
- Modelos de Métodos Formais
- Modelos Evolutivos de Processo de Software
  - Modelo Incremental.
  - Modelo Espiral.
  - Modelo de Montagem de Componentes.
  - Modelo de Desenvolvimento Concorrente.

Modelo de Desenvolvimento Concorrente

- Também chamado de **engenharia concorrente**.
- Possibilita à equipe de software representar elementos concorrentes e iterativos de qualquer um dos modelos de processo explanados anteriormente.

Modelo de Desenvolvimento Concorrente



Modelo de Desenvolvimento Concorrente

- A modelagem concorrente define uma série de eventos que vão disparar transições de um estado para outro para cada uma das atividades, ações ou tarefas da engenharia de software.
- Por exemplo, durante os estágios iniciais do projeto (uma ação de engenharia de software importante que ocorre durante a atividade de modelagem), uma inconsistência no modelo de requisitos não é descoberta.
- Isso gera o evento *correção do modelo de análise*, que vai disparar a ação de análise de requisitos, passando do estado **concluído** para o estado **aguardando modificações**.

Modelo de Desenvolvimento Concorrente

- A modelagem concorrente se aplica a todos os tipos de desenvolvimento de software e fornece uma imagem precisa do estado atual de um projeto.
- Em vez de limitar as atividades, ações e tarefas da engenharia de software a uma sequência de eventos, ela define uma rede de processos.
- Cada atividade, ação ou tarefa na rede existe simultaneamente com outras atividades, ações ou tarefas.

Modelos de Processo de Software

- Modelo Sequencial Linear
  - Também chamado Ciclo de Vida Clássico ou Modelo Cascata.
- O Paradigma de Prototipação
- Técnicas de Quarta Geração
- Modelo RAD (Rapid Application Development)
- Modelos de Métodos Formais
- Modelos Evolutivos de Processo de Software
  - Modelo Incremental.
  - Modelo Espiral.
  - Modelo de Montagem de Componentes.
  - Modelo de Desenvolvimento Concorrente.

Modelos de Métodos Formais

- Inclui um conjunto de atividades que conduzem à especificação matemática formal do software.
- Permitem especificar, desenvolver e verificar um sistema baseado em computador pela aplicação de uma notação matemática rigorosa.
- O uso de métodos formais durante o desenvolvimento oferece um mecanismo de eliminação de muitos dos problemas difíceis de serem superados com o uso de outros paradigmas de engenharia de software.

Modelos de Métodos Formais

- Ambiguidade, incompletude e inconsistência podem ser descobertas e corrigidas mais facilmente – não por meio de uma revisão local, mas devido à aplicação de análise matemática.
- Quando são utilizados métodos formais durante o projeto, eles servem como base para verificar a programação e, portanto, possibilitam a descoberta e a correção de erros que, de outra forma, poderiam passar despercebidos.

Modelos de Métodos Formais

Desvantagem:

- Atualmente, o desenvolvimento de modelos formais consome muito tempo e dinheiro.
- Como poucos desenvolvedores de software possuem formação e experiência necessárias para aplicação dos métodos formais, é necessário treinamento extensivo.
- É difícil usar os modelos como meio de comunicação com clientes tecnicamente despreparados (não sofisticados tecnicamente).

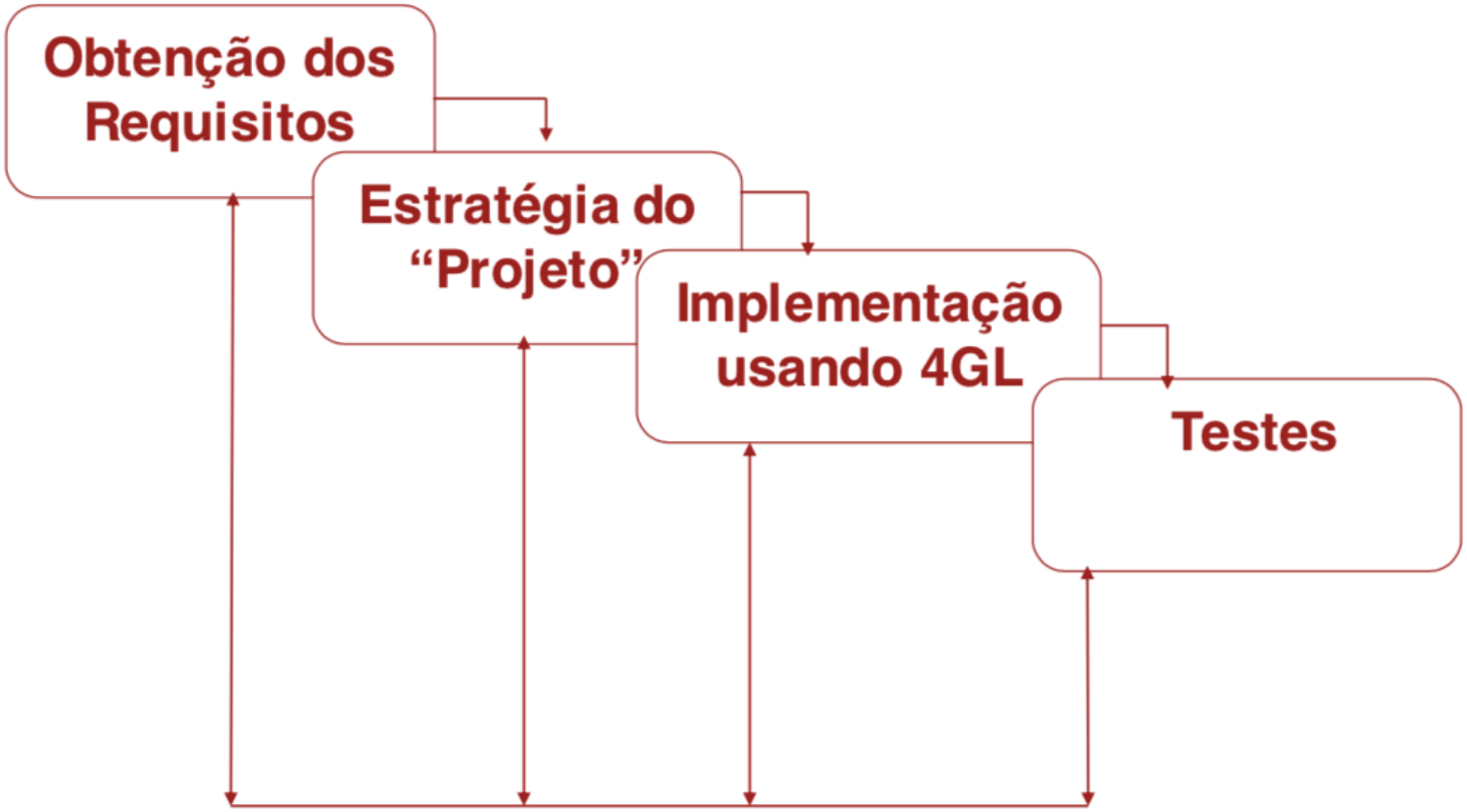
Técnicas de 4ª Geração

- Concentra-se na capacidade de se especificar o software a uma máquina em um nível que esteja próximo à linguagem natural.
- Engloba um conjunto de ferramentas de software que possibilitam que:
  - O sistema seja especificado em uma linguagem de alto nível.
  - O código fonte seja gerado automaticamente a partir dessas especificações

Técnicas de 4ª Geração

- O ambiente de desenvolvimento de software que sustenta o ciclo de vida de 4ª geração inclui as ferramentas:
  - Linguagens não procedimentais para consulta de banco de dados;
  - Geração de relatórios;
  - Manipulação de dados;
  - Interação e definição de telas;
  - Geração de códigos;
  - Capacidade gráfica de alto nível;
  - Capacidade de planilhas eletrônicas.

Técnicas de 4ª Geração



Técnicas de 4ª Geração

Obtenção dos Requisitos

- O cliente descreve os requisitos os quais são traduzidos para um protótipo operacional.
- O cliente pode estar inseguro quanto aos requisitos.
- As 4GLs atuais não são sofisticadas suficientemente para acomodar a verdadeira "linguagem natural".
- O cliente pode ser incapaz de especificar as informações de um modo que uma ferramenta 4GL possa consumir

Técnicas de 4ª Geração

Estratégia do Projeto

- Para pequenas aplicações é possível mover-se do passo de Obtenção dos Requisitos para o passo de Implementação usando uma linguagem de quarta geração.
- Para grandes projetos é necessário desenvolver uma estratégia de projeto. De outro modo ocorrerão os mesmos problemas encontrados quando se usa abordagem convencional (baixa qualidade).

Técnicas de 4ª Geração

Implementação Usando 4GL

- Os resultados desejados são representados de modo que haja geração automática de código. Deve existir uma estrutura de dados com informações relevantes e que seja acessível pela 4GL.

Testes

- O desenvolvedor deve efetuar testes e desenvolver uma documentação significativa. O software desenvolvido deve ser construído de maneira que a manutenção possa ser efetuada prontamente.

Técnicas de 4ª Geração

Comentários

- Vantagens
  - Redução dramática no tempo de desenvolvimento do software (aumento de produtividade).
- Desvantagens:
  - As 4GL atuais não são mais fáceis de usar do que as linguagens de programação.
  - O código fonte produzido é ineficiente.
  - A manutenibilidade de sistemas usando técnicas 4G ainda é questionável.

Qual modelo de processo de software escolher?

- Depende:
  - Natureza do projeto e da aplicação.
  - Métodos e ferramentas a serem usados.
  - Controles e produtos que precisam ser entregues.