

# Disciplina: Engenharia de Software

## Prof. Adriano Nakamura

Baseado no material do Prof. Caio Saraiva Coneglian



# Objetivo

Objetivo da aula é apresentar os modelos clássicos de desenvolvimento de software.

# Modelo Genérico

# Modelo Genérico

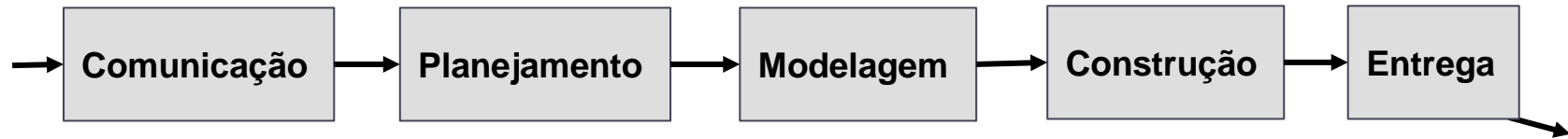
- Cinco atividades:

- Comunicação;
- Planejamento;
- Modelagem;
- Construção;
- Entrega.

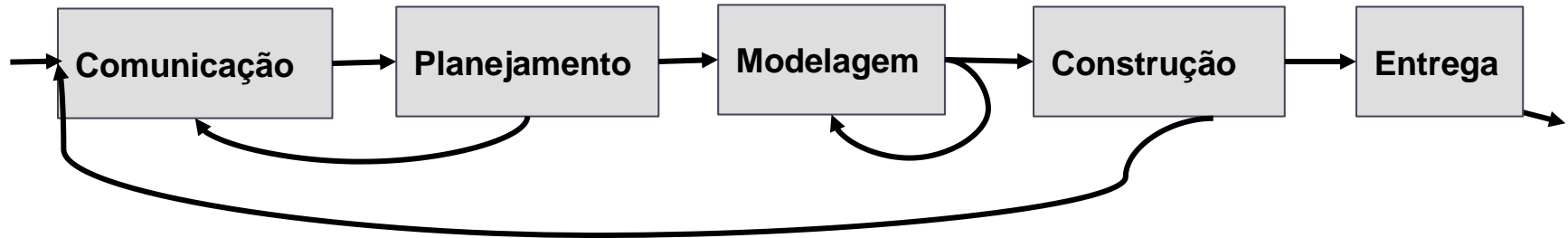
- Fluxo:

- Linear;
- Iterativo;
- Evolucionário;
- Paralelo

# Fluxo dos Modelos Genéricos

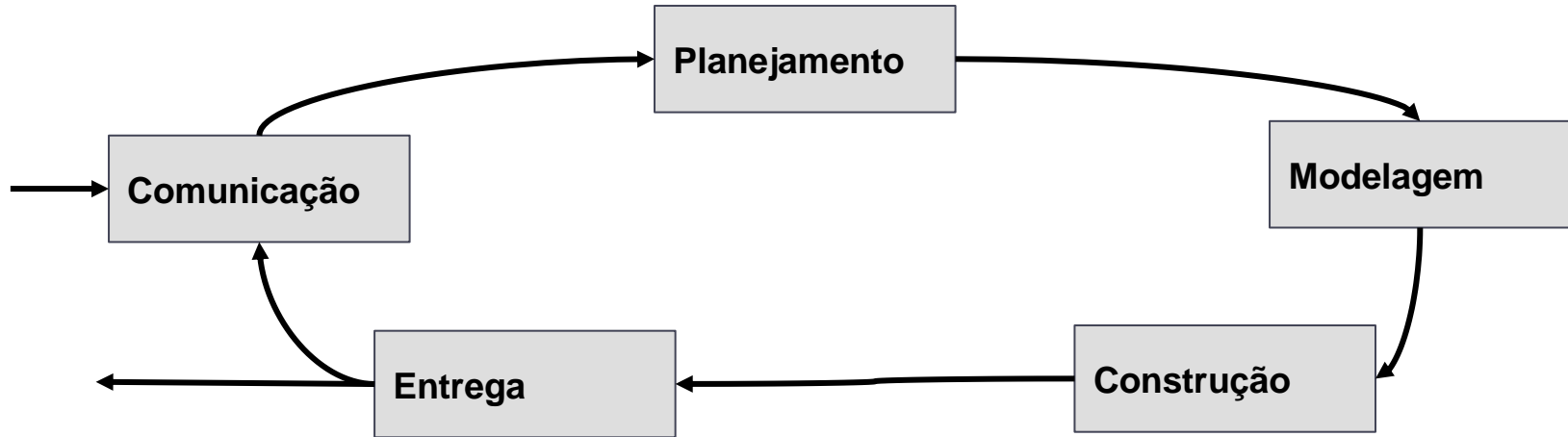


**Fluxo de Processo linear**



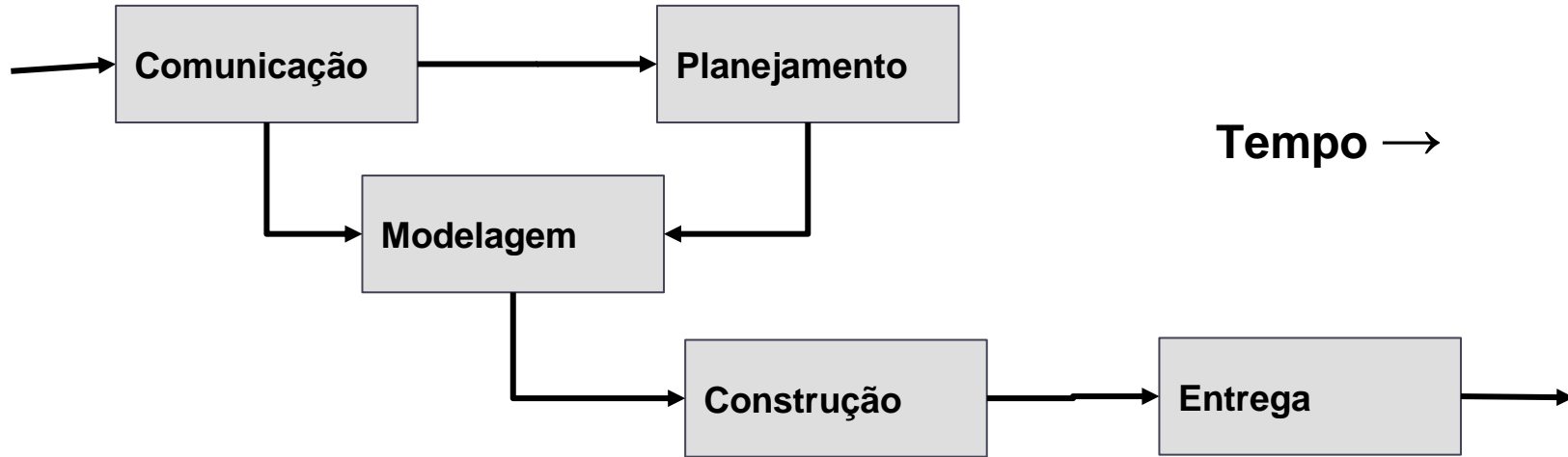
**Fluxo de Processo iterativo**

# Fluxo dos Modelos Genéricos



**Fluxo de Processo evolucionário**

# Fluxo dos Modelos Genéricos



**Fluxo de Processo paralelo**

# Partindo desses Processos

- Os modelos apresentados são genéricos:
  - Ou seja:
    - Não são utilizados na prática.
- A partir deles:
  - Há diversos Modelos Reais.



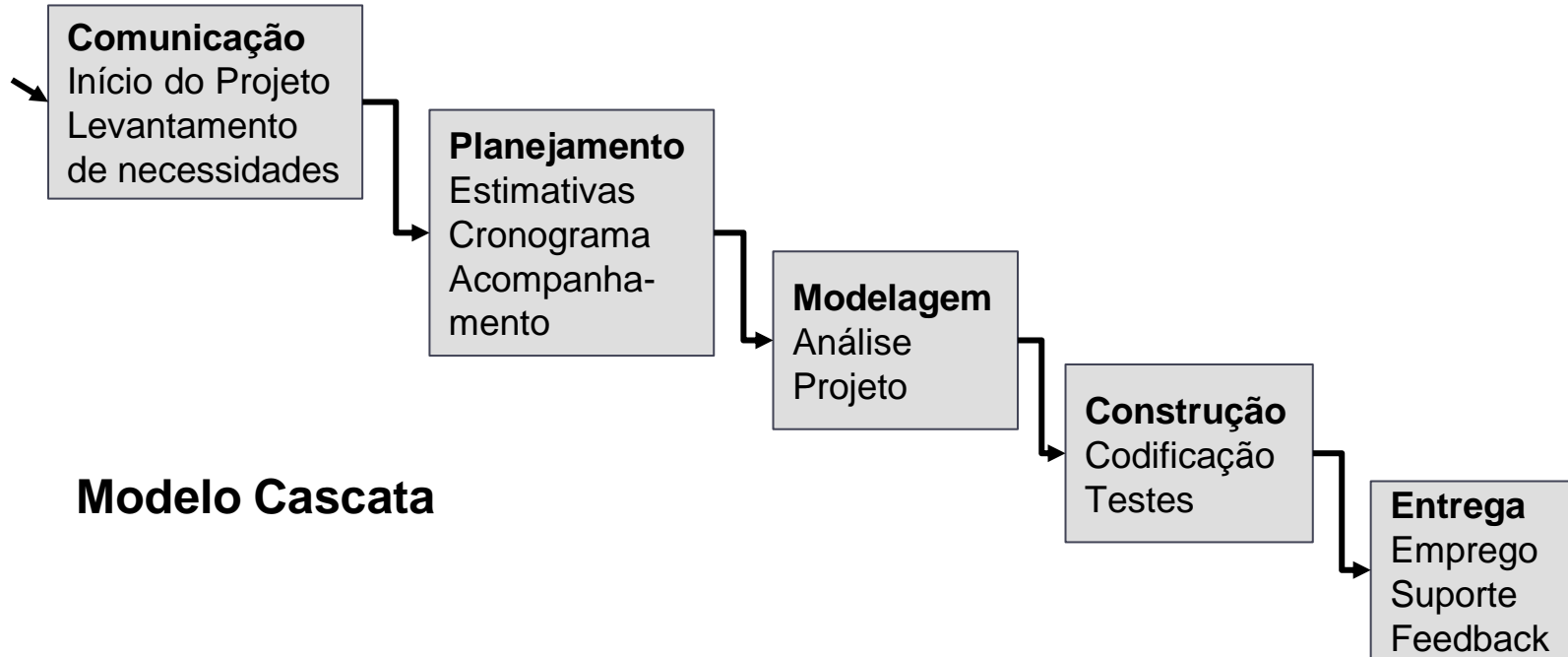
# Modelo Cascata



# Modelo Cascata

- Conhecido como **Ciclo de Vida Clássico**;
- Um tipo de modelo prescritivo;
- Modelo tradicional;
- Flui da **Comunicação** ao **Emprego** de forma linear;
- Muito utilizado;

# Modelo Cascata



# Modelo Cascata

- Sequencial e sistemática;
  - Levantamento das necessidades;
  - Planejamento;
  - Modelagem;
  - Construção;
  - Emprego;
  - Suporte ao software concluído.

# Modelo Cascata

- Bastante rígido;
  - Não pode-se ir e voltar pelas fases;
  - Somente pode-se avançar e totalmente uma parte.



# Problema do Modelo Cascata

- A realidade normalmente não é sequencial;
- Difícil estabelecer todos os requisitos no início do projeto;
- Requisitos mal descritos pelo cliente, mal entendido pelo analista;
- Mudança de cenário na organização que exija adaptação de requisitos;
- O modelo em cascata não prevê revisão de fases;
- Demora para se ter uma versão executável, o cliente deve ter paciência.

# Quando usar?

- Quando está bem definido os requisitos;
- Cliente e equipe compreende bem o que quer e onde deseja chegar:
  - Não haverá mudanças de requisitos.

# Modelo V

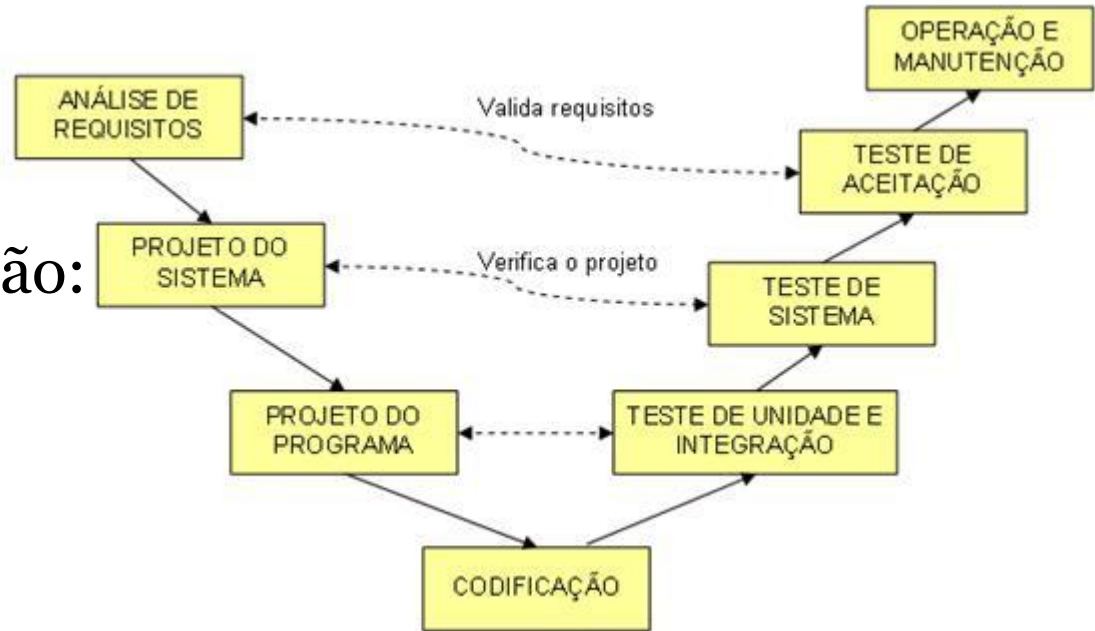


# Modelo V

- Elaborado pelo Ministério de Defesa da Alemanha, 1992;
- Variação do Modelo Cascata;
- Segue **a ideia sequencial**;
- Porém considera aspectos para as fases finais.
  - Faz pensando nas próximas fases;
  - Fases anteriores olham para as fases passadas.

# Modelo V

- Ênfase dada à verificação e validação: cada fase do lado esquerdo gera um plano de teste a ser executado no lado direito.



# Modelo V

- Foco na Qualidade do Software;
  - Tudo que eu faço, eu penso no seu teste;
  - Tudo será testado:
    - Desde a modelagem, até:
    - Os códigos.

# Modelo V e Cascata

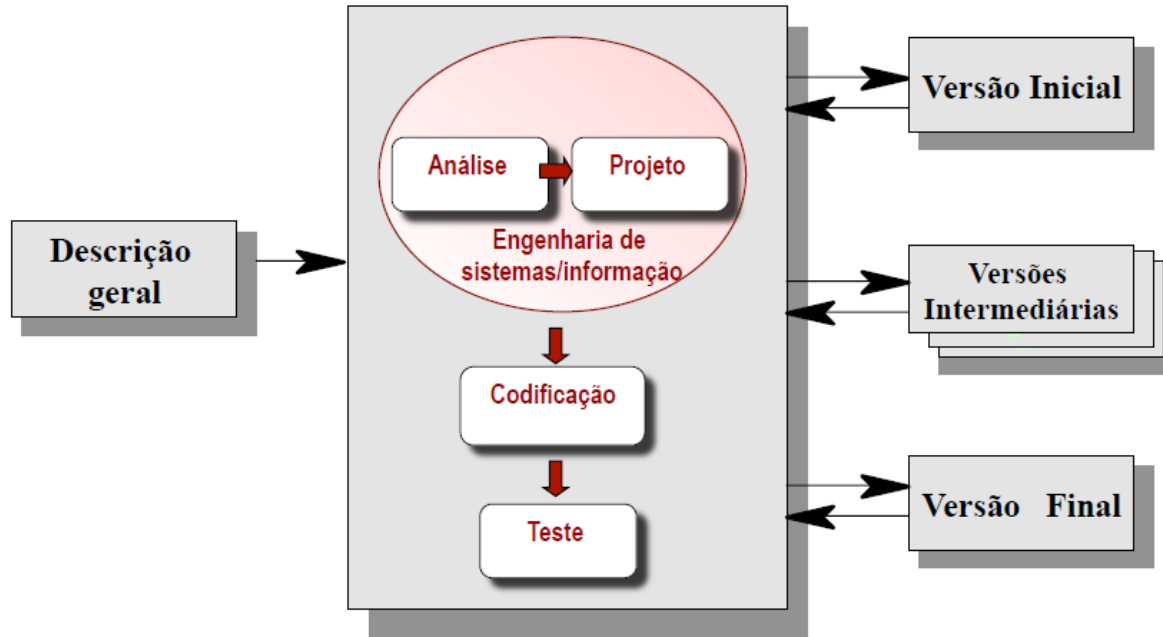
- Modelo V é um modelo Cascata:
  - Pequena variações.
- **Tem-se os mesmos problemas no Modelo V que no modelo Cascata.**
- Da mesma forma que o modelo em cascata, o cliente só recebe a primeira versão do software no final do ciclo, mas **apresenta menos risco**, devido ao planejamento prévio dos testes nas fases de análise e projeto.

# Modelo Incremental

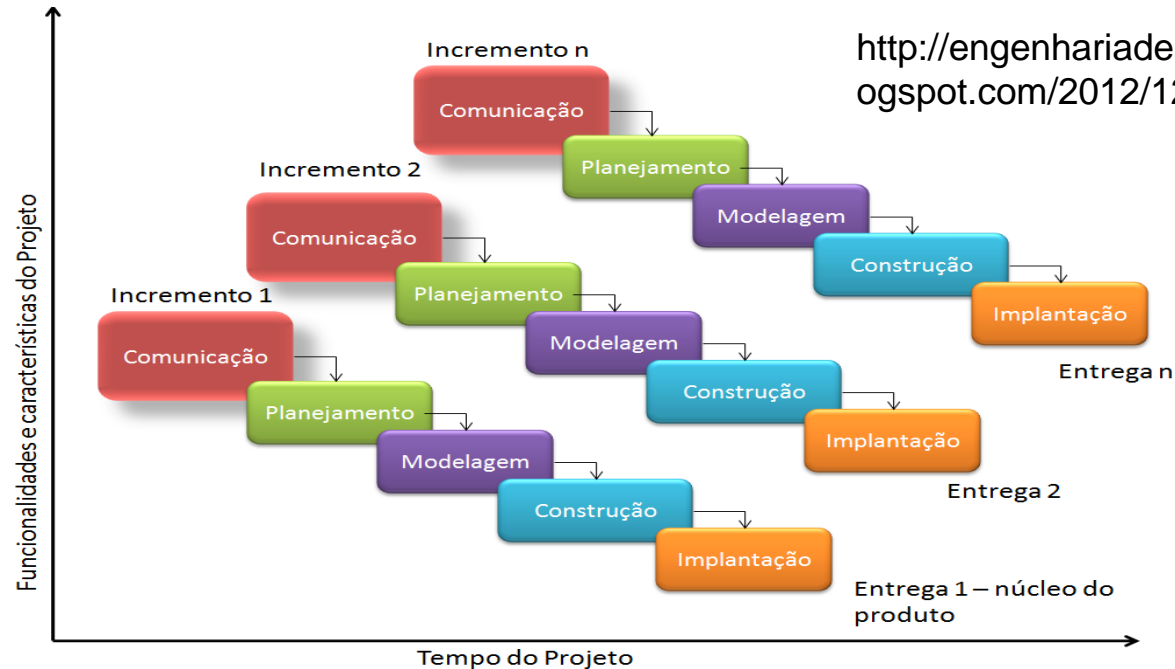
# Modelo Incremental

- Os requisitos do cliente são obtidos, e, de acordo com a funcionalidade, são agrupados em módulos.
- Após este agrupamento, a equipe, junto ao cliente, define a prioridade em que cada módulo será desenvolvido, escolha baseada na importância daquela funcionalidade ao negócio do cliente.
- Cada módulo passará por todas as fases "cascata" de projeto e será entregue ao cliente um software operacional.
- Assim, o cliente receberá parte do produto final em menos tempo.

# Modelo Incremental



# Modelo Incremental



<http://engenhariadesoftwareuesb.blogspot.com/2012/12/blog-post.html>



# Quando utilizar?

- Requisitos bem definidos;
  - Não precisa ir e voltar;
- Mas o cliente necessita de uma versão.
  - Entregar algo rapidamente.



# 1º Incremento

- Normalmente:
  - O 1º incremento é um software funcional;
  - Cliente:
    - Avalia essa 1ª versão;
    - Se planeja uma segunda versão;
  - Planejamento>
    - Já considera que haverá mudanças;
    - Mas já tem um plano do que será desenvolvido.

# Exemplo



- Software de Processamento de Texto:
  - 1º Incremento: produto essencial. Gerenciamento de arquivos, edição e produção de documentos;
    - Requisitos básicos são atendidos
  - 2º Incremento: recursos mais sofisticados de edição e de produção;
  - 3º Incremento: Revisão ortográfica e gramatical;
  - 4º Incremento: Recursos avançados de formatação (layout) de página.
    - Produto completo atendendo as solicitações.

# Modelo Evolucionário

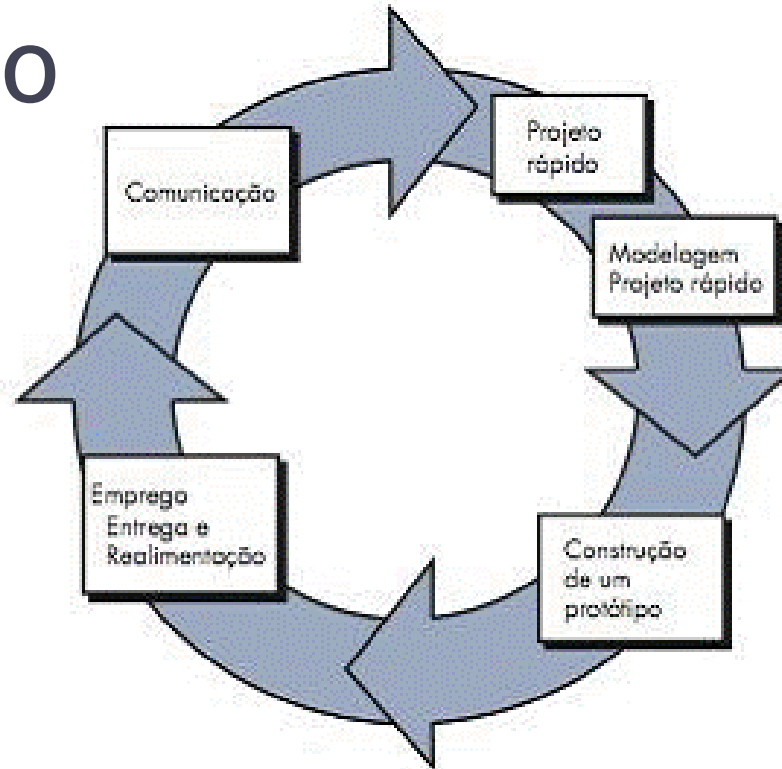
# Modelo Evolucionário

- Fluxos iterativos;
- Consideram a evolução ao longo do tempo;
- Muitas vezes há dificuldade de se cumprir os prazos;
- Modelos:
  - Prototipação;
  - Modelo Espiral.

# Prototipação

- Quando não há detalhes dos requisitos para funções e recursos;
- Ou, desenvolvedor inseguro quanto à eficiência de um algoritmo;
- Iterações;
- Normalmente a primeira versão não é utilizável;
- Ter uma ideia prévia do sistema.

# Prototipação



# Prototipação

- Comunicação:
  - Obter os requisitos;
- Projeto Rápido e Modelagem:
  - Projetar aquilo que é visível ao usuário - I/O;
- Construção:
  - Implementar isso;
- Emprego:
  - Entrega, avaliação e retroalimentação;
- Começa novamente refinando o protótipo.



Importante

**Protótipo é utilizado para identificar  
REQUISITOS**

**Posteriormente, pode ser feito um  
protótipo FUNCIONAL**

# Interessante

- Usuários:
  - Tem uma prévia do sistema final;
- Desenvolvedores e engenheiros:
  - Passam a desenvolver de imediato.

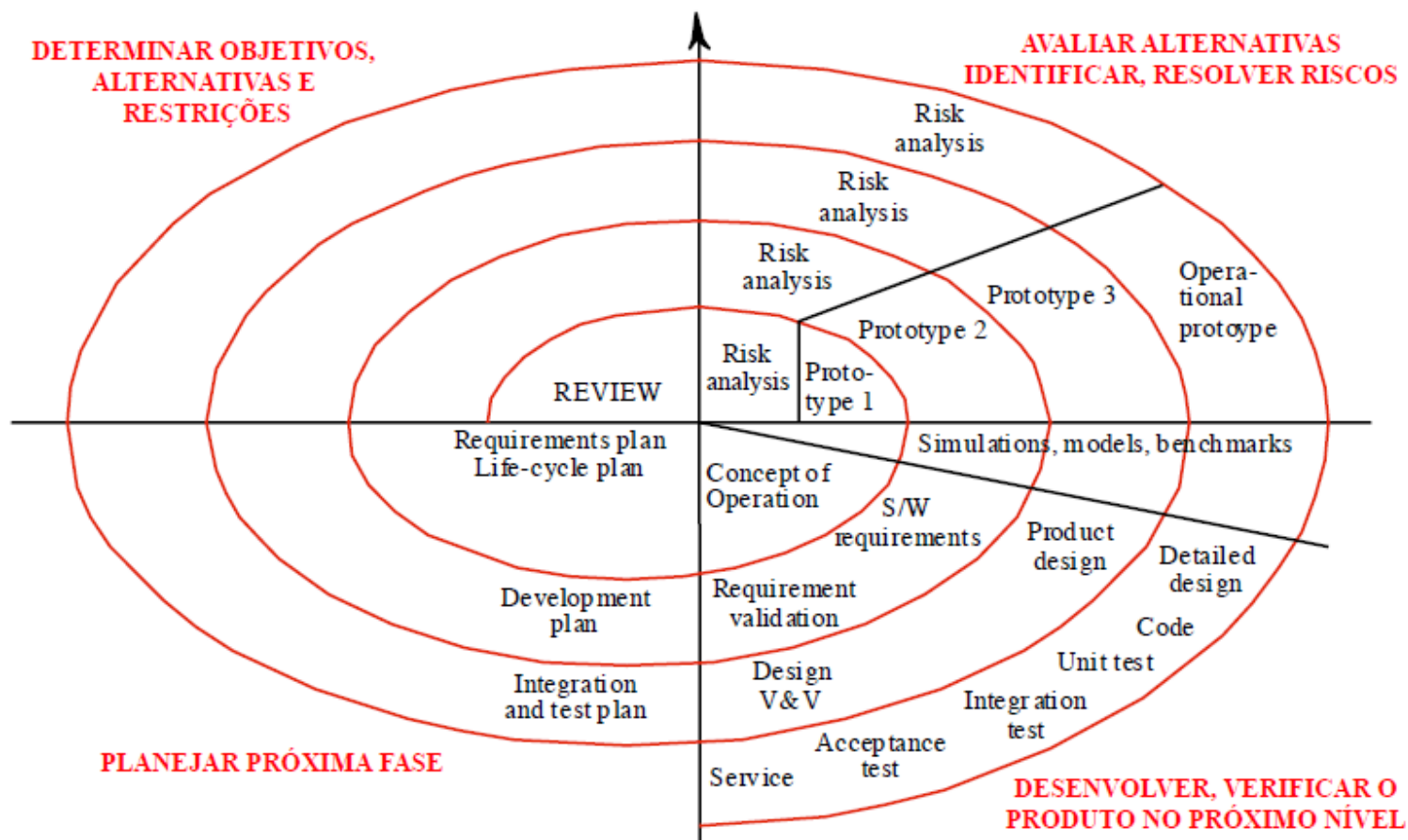


# Prototipação

- Pode ser problemático:
  - Clientes acharem que é **versão utilizável**;
    - Acham que vão vir poucas correções.
    - **NÃO!!!**
  - Os engenheiros de software assumem um compromisso de colocar um sistema viável rápido, a partir do protótipo.
    - Pode **diminuir a qualidade**.

# Modelo Espiral

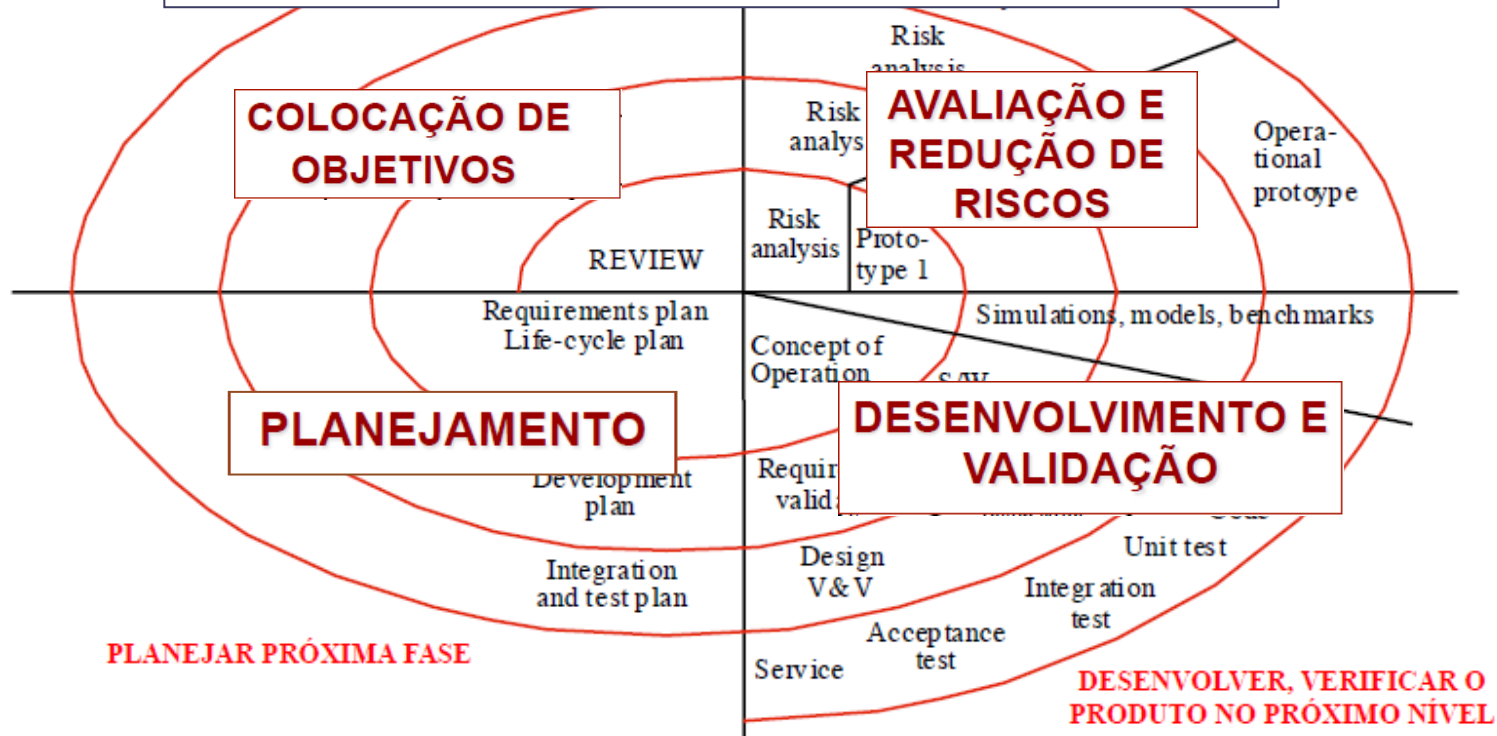
- Engloba as melhores características do ciclo de vida Clássico e da Prototipação, adicionando um novo elemento: a *Análise de Risco*
- Segue a abordagem de passos sistemáticos do Ciclo de Vida Clássico incorporando-os numa estrutura *iterativa* que reflete mais realisticamente o mundo real
- Usa a *Prototipação*,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



DETERM  
ALTE  
RE

➤ Cada “loop” do espiral é dividido em 4 setores

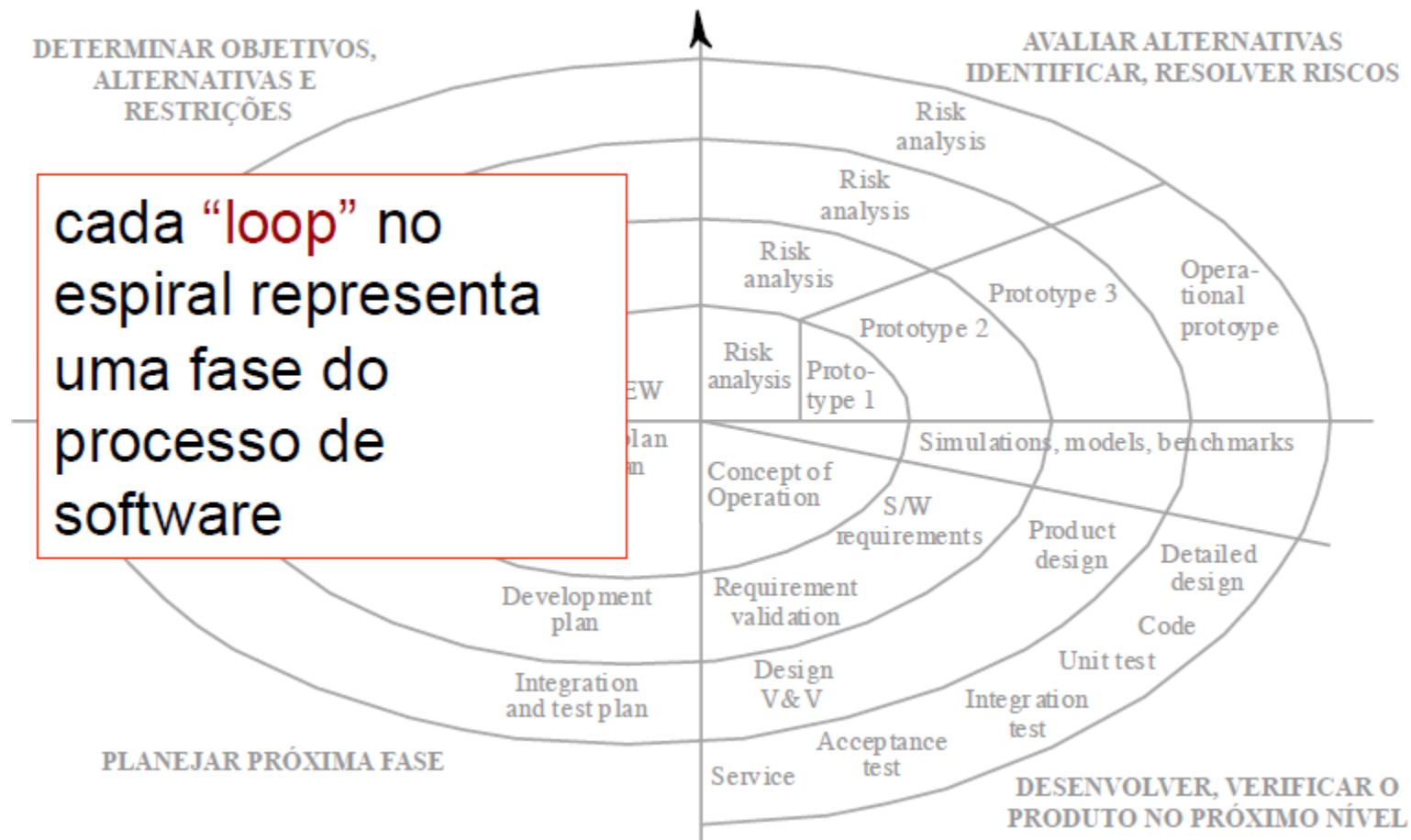
ALTERNATIVAS  
OLVER RISCOS



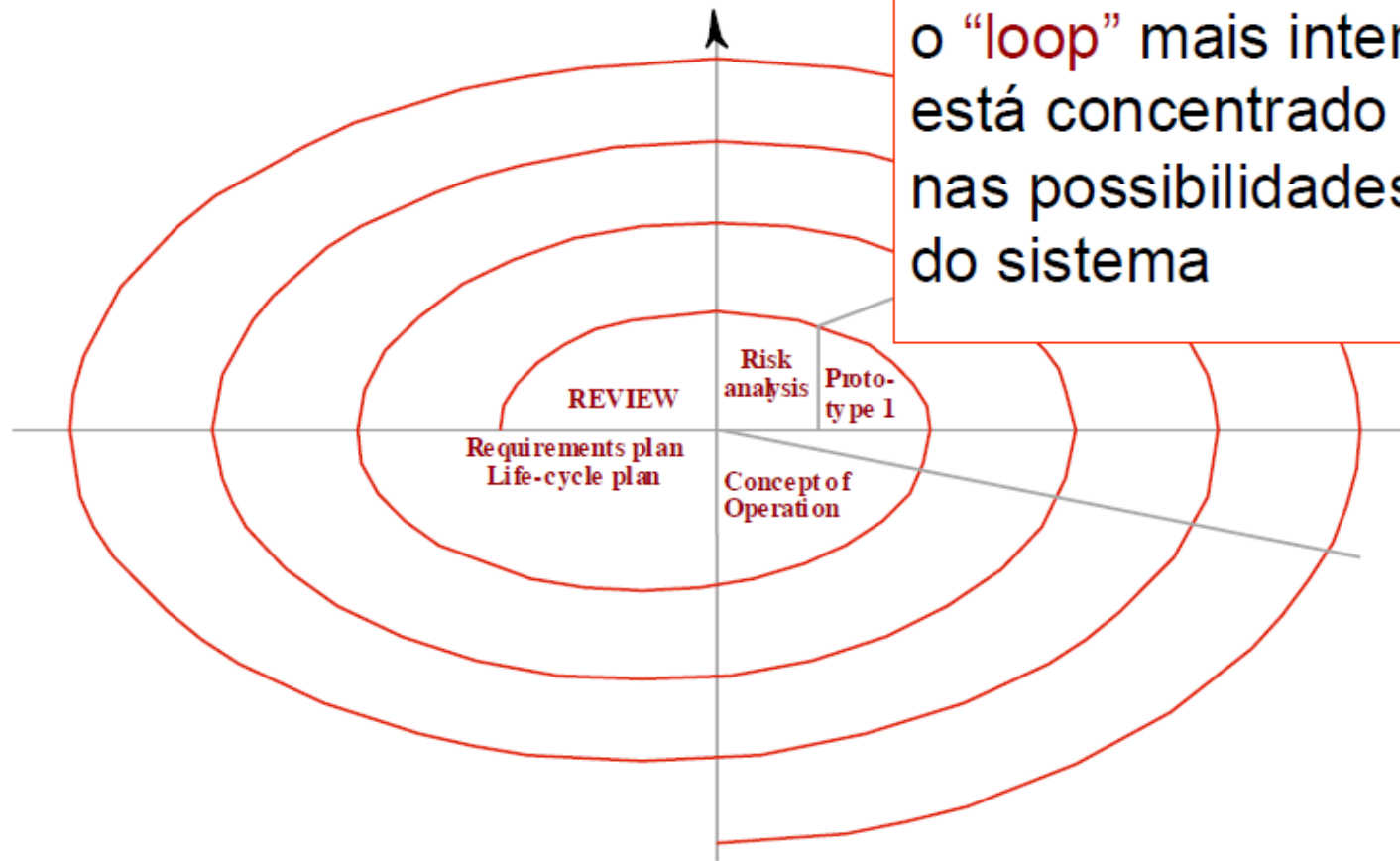
# Modelo Espiral

- **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,
- **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
- **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” )

cada “loop” no  
espiral representa  
uma fase do  
processo de  
software

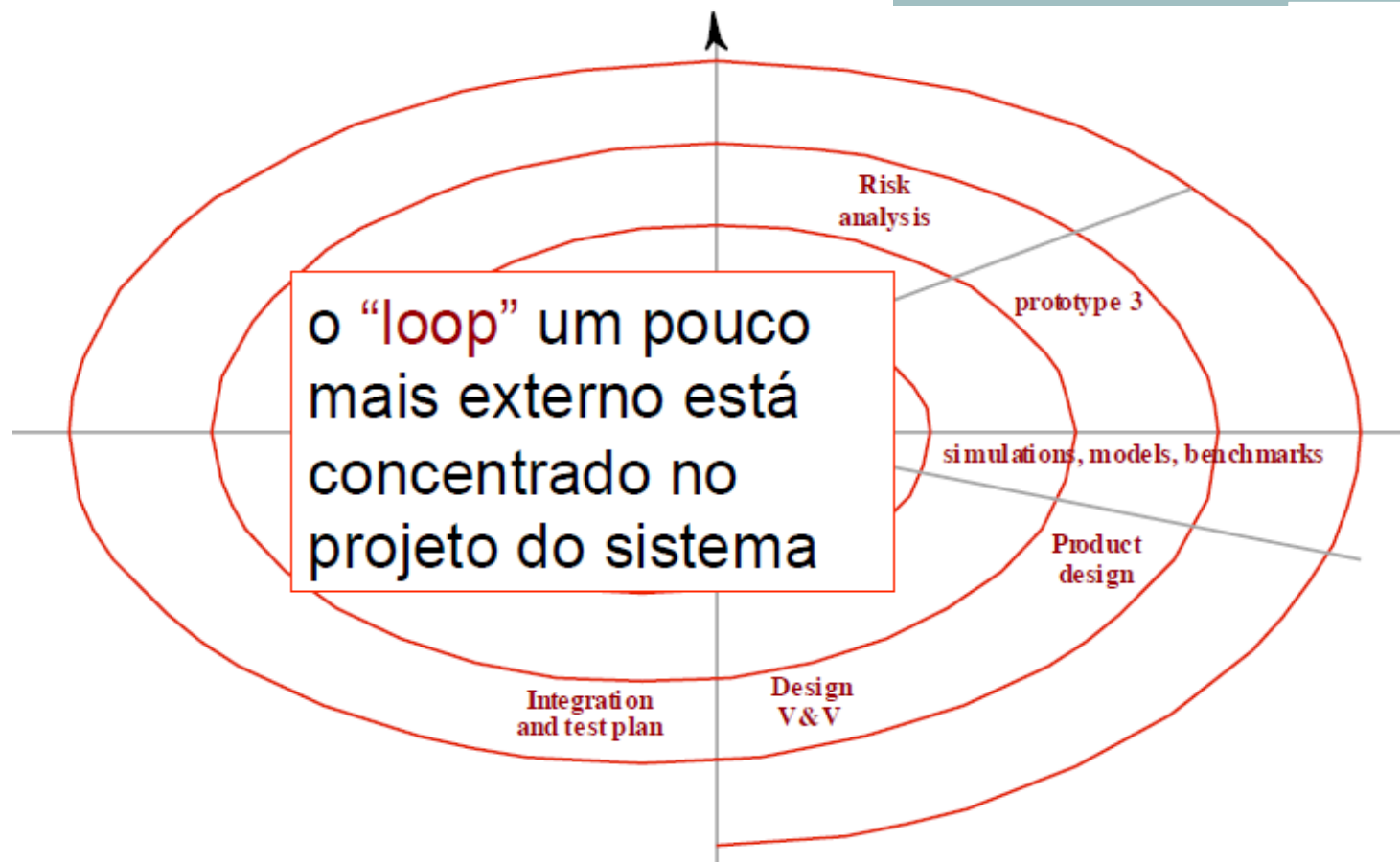


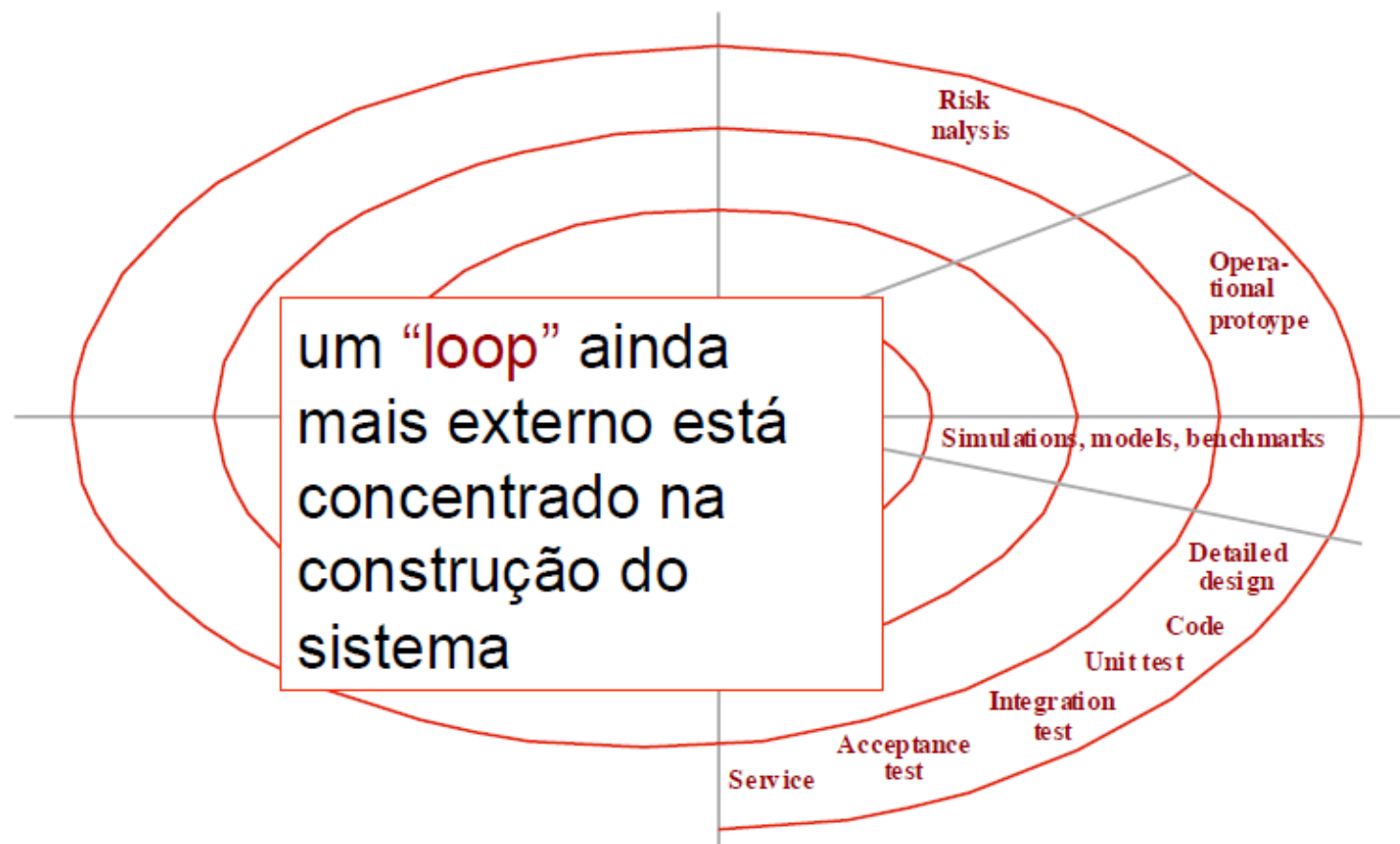




o próximo “loop”  
está concentrado na  
definição dos  
requisitos do  
sistema







DETERMINAR OBJETIVOS,  
ALTERNATIVAS E  
RESTRICÇÕES

AVALIAR ALTERNATIVAS  
IDENTIFICAR, RESOLVER RISCOS

Risk

- não existem fases fixas no modelo
- as fases mostradas na figura são meramente exemplos
- a gerência decide como estruturar o projeto em fases

Integration  
and test plan

Design  
V&V

Unit test

Integration  
test

Acceptance  
test

Service

PLANEJAR PRÓXIMA FASE

DESENVOLVER, VERIFICAR O  
PRODUTO NO PRÓXIMO NÍVEL

# Processo Unificado

# Processo Unificado

- Utilizar os princípios clássicos do desenvolvimento de software;
- Considerando os princípios do desenvolvimento ágil;
- Fluxo de processo iterativo e incremental;
- Sensação evolucionária

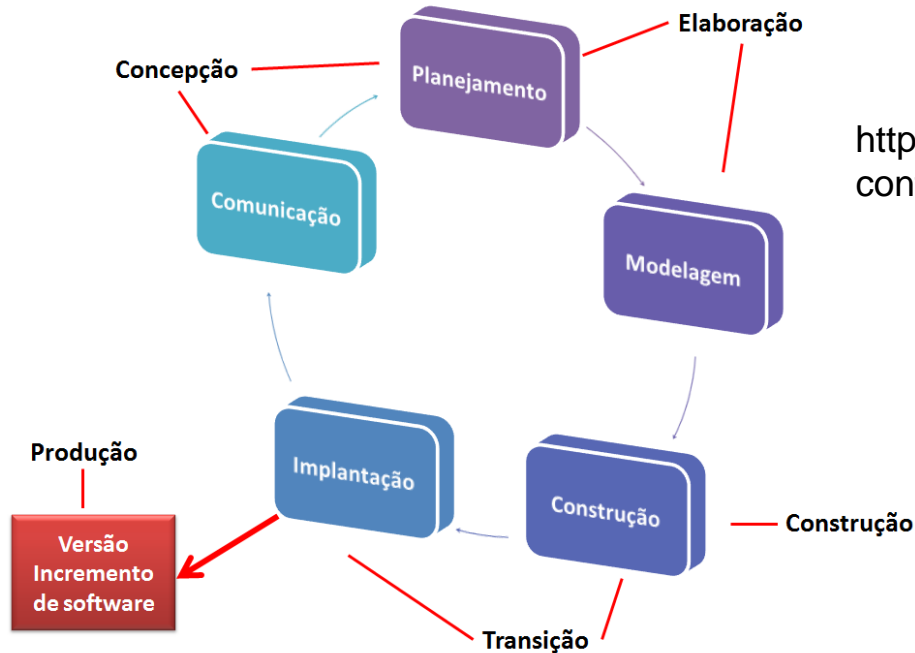


# Processo Unificado

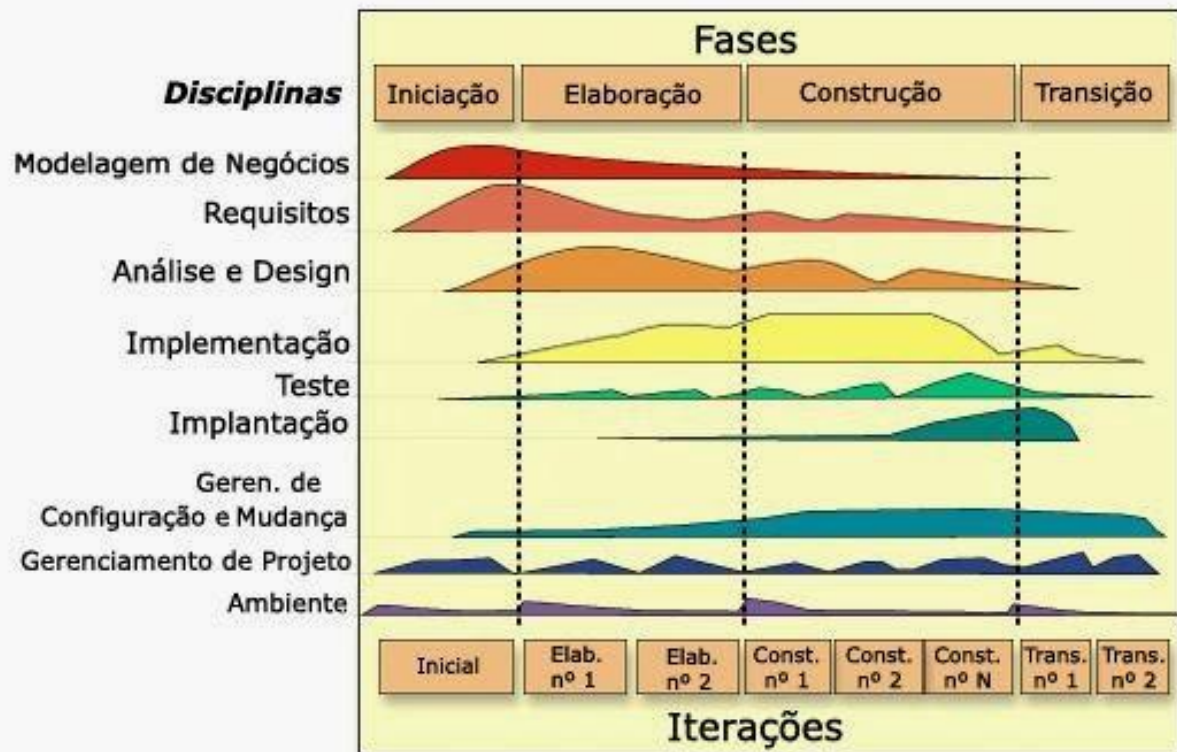
- **Direcionado por casos de uso:** O início do processo deve ser marcado pela utilização dos casos de uso, a fim de se definir uma linguagem entre os usuários e o sistema, facilitando a especificação dos requisitos.
- **Centrado na arquitetura:** O processo procura modelar uma arquitetura através dos aspectos estáticos e dinâmicos de um projeto, que podem ser obtidos junto a um estudo direcionado pelos casos de uso mais significativos.
- **É iterativo e incremental:** Uma das práticas do processo é dividir grandes projetos em mini-projetos. Cada mini-projeto possui uma iteração, que quase sempre abrange todo o fluxo de trabalho. Olhando como um todo, essa iteração resulta em um incremento para o projeto. É válido lembrar que as iterações são planejadas de acordo com os casos de uso.



# Processo Unificado - Fases



<http://jkolb.com.br/wp-content/uploads/2013/12/up.png>



## Iniciação

**Essa fase tem como objetivos preparar o ambiente de suporte, estabelecer o escopo do projeto, determinar os casos de uso, estimar o custo e prazo totais e identificar os riscos potenciais. O principal artefato dessa fase é o Documento de Visão.**

→ **Marco: Objetivo do Ciclo de vida**

O marco do objetivo do ciclo de vida é o que avalia e diz sobre a viabilidade inicial do projeto.

## Elaboração

**A fase de elaboração tem como meta criar a baseline de arquitetura do sistema, cujo objetivo é fornecer uma base estável para a construção. Primeiramente é realizado um exame dos requisitos mais significativos, ou seja, os que têm um maior impacto na arquitetura. Durante esta fase são realizados protótipos de arquitetura a fim de que seja verificada a estabilidade da arquitetura escolhida.**

**→ Marco: Arquitetura do ciclo de vida**

O marco desta fase é a arquitetura do ciclo de vida que pode ser verificada através do documento de arquitetura. É durante esta fase que se define uma baseline gerenciada para a arquitetura do software, através desta definição é que o escalonamento da equipe ocorre na fase seguinte de Construção.

## Construção

**A fase de construção é literalmente de manufatura, em que o foco é no gerenciamento de recursos e controle de operação, visando a otimização de custos, programação e um produto de qualidade. Tem como meta concluir o desenvolvimento e esclarecer possíveis requisitos restantes das fases anteriores. Deixa-se o campo do desenvolvimento intelectual e parte para o ataque das áreas de desenvolvimento do produto, para que possa ser implementado durante a construção e transição.**

**→ Marco: Capacidade Operacional Inicial**

O marco da fase de construção é certificar se o produto está pronto para ser implementado em um ambiente em teste beta.

## Transição

**A fase de transição do RUP é basicamente a fase em que é feita a entrega do produto ao cliente, ou seja, ela tem o objetivo de assegurar que o software produzido esteja disponível aos usuários finais.**

**→ Marco: Entrega do sistema para o cliente**

O marco da fase de transição é certificar se o produto está pronto para ser instalado e mantido pelo cliente.

# Disciplina de Modelagem de Negócios

Modelagem de negócios, explica como descrever uma visão da organização na qual o sistema será implantado e como usar esta visão como uma base para descrever o processo, papéis e responsabilidades.



## **Modelagem de Negócios**

Compreender o negócio significa entender as necessidades e os problemas do cliente.

# Disciplina de Requisitos

Esta disciplina explica como levantar pedidos das partes interessadas ("stakeholders") e transformá-los em um conjunto de requisitos que visam estabelecer uma concordância com os clientes e outros envolvidos com sistema.



## **Disciplina de Requisitos**

Interface além do  
planejamento de custos  
relacionados ao projeto



# Disciplina de Análise e Projeto("Design")

Esta disciplina busca levantar e mostrar o que será desenvolvido no  
Quais as necessidades e como resolvê las de uma forma lógica.  
Podendo ser utilizado como um gabarito para o desenvolvimento.



## **Disciplina de Análise de Projeto**

Executado, em um ambiente de execução determinado, as tarefas e funções especificadas nas descrições de casos de uso.

# Disciplina de Teste

Sistemas são criados através da aplicação de componentes. O processo Descreve como reutilizar componentes existentes ou implementar novos Componentes com responsabilidade bem definida, tornando o sistema Mais fácil de manter e aumentando as possibilidades de reutilização.



## **Disciplina de Teste**

Verificar as interações, integrações, requisitos, e evitar os defeitos dos objetos e componentes do software.

# Disciplina de Implantação

O objetivo da Implantação é o de produzir com sucesso lançamentos de produtos e entregar o software para seus usuários finais.



## Disciplina de Implantação

Os processos ("workflows") de "Implantação e Ambiente" do RUP contém menos detalhes do que outros *workflows*.

# Disciplina de Ambiente

A proposta das atividades de ambiente é prover à organização de desenvolvimento de software os processos e as ferramentas que darão suporte à equipe de desenvolvimento.




## **Disciplina de Ambiente**

O ambiente enfoca as atividades necessárias para configurar o processo para um projeto.

# Disciplina de Configuração e Gerência de Mudança

A disciplina de Gestão de Mudança em negócios com RUP abrange três gerenciamentos específicos: de configuração, de solicitações de mudança, e de status e medição.



## **Disciplina de Configuração e Gerência de Mudança**

Responsável por organizar, planejar, avaliar, executar e testar as mudanças realizadas no software.

# Disciplina de Gerência de Projeto

O planejamento de projeto no RUP ocorre em dois níveis. Há uma baixa granularidade ou planos de Fase que descreve todo o projeto, e uma série de alta granularidade ou planos de Iteração que descrevem os passos iterativos.



## **Disciplina de Gerência de Projeto**

Outros papéis são:  
Gestão de pessoas,  
gestão de custos e  
fornecedores

# Comparação entre os Modelos

| Modelo             | Foco                       | Requisitos                              | 1ª versão p/ cliente                        | Gerenciamento<br>(1=mais simples) | Complexidade<br>do Sistema |
|--------------------|----------------------------|---|---|-----------------------------------|----------------------------|
| Cascata            | Documento e artefato       | Bem conhecido/congelado                 | Fim do ciclo                                | 1                                 | Simples                    |
| V                  | Planejamento de testes     | Bem conhecido/congelado                 | Fim do ciclo                                | 2                                 | Simples                    |
| Incremental        | Incrementos operacionais   | Maior abstração / Tratado em módulos    | Protótipos operacionais                     | 3                                 | Médio                      |
| Evolucionário      | Evolução dos requisitos    | Pouco conhecidos                        | Protótipos operacionais                     | 4                                 | Médio                      |
| Prototipação       | Dúvidas nos Requisitos     | Abstratos                               | Protótipos não operacionais                 | 5                                 | Médio                      |
| Espiral            | Análise de risco           | Maior abstração / evoluídos com o tempo | Protótipos operacionais ou não operacionais | 5                                 | Complexos                  |
| Processo Unificado | Frameworks e boas práticas | Maior abstração / evoluídos com o tempo | Protótipos operacionais ou não operacionais | 5                                 | Complexos                  |