

MODELOS DE PROCESSO DE SOFTWARE

Definição e Elementos de Processo de Software
Alguns Modelos Prescritivos
Alguns Modelos Ágeis
Visão Genérica do Desenvolvimento de Software

Modelos de Processo de Software

- Procuram descrever formalmente e de maneira organizada todas as atividades que devem ser seguidas para a obtenção segura de um produto de software
- A escolha do modelo de processo de software depende:
 - da **natureza** do projeto e da aplicação
 - dos **métodos** e **ferramentas** a serem usados
 - dos **controles** e **produtos** que precisam ser entregues
- Existem vários *modelos de processo de software* (ou *paradigmas de engenharia de software* ou *modelos de ciclo de vida*)
- Cada um representa uma tentativa de colocar ordem em uma atividade inerentemente caótica

Modelos de Processo de Software

Alguns Modelos de Processos Prescritivos Tradicionais

 Modelo Seqüencial Linear (Modelo Cascata)

 Paradigma de Prototipação

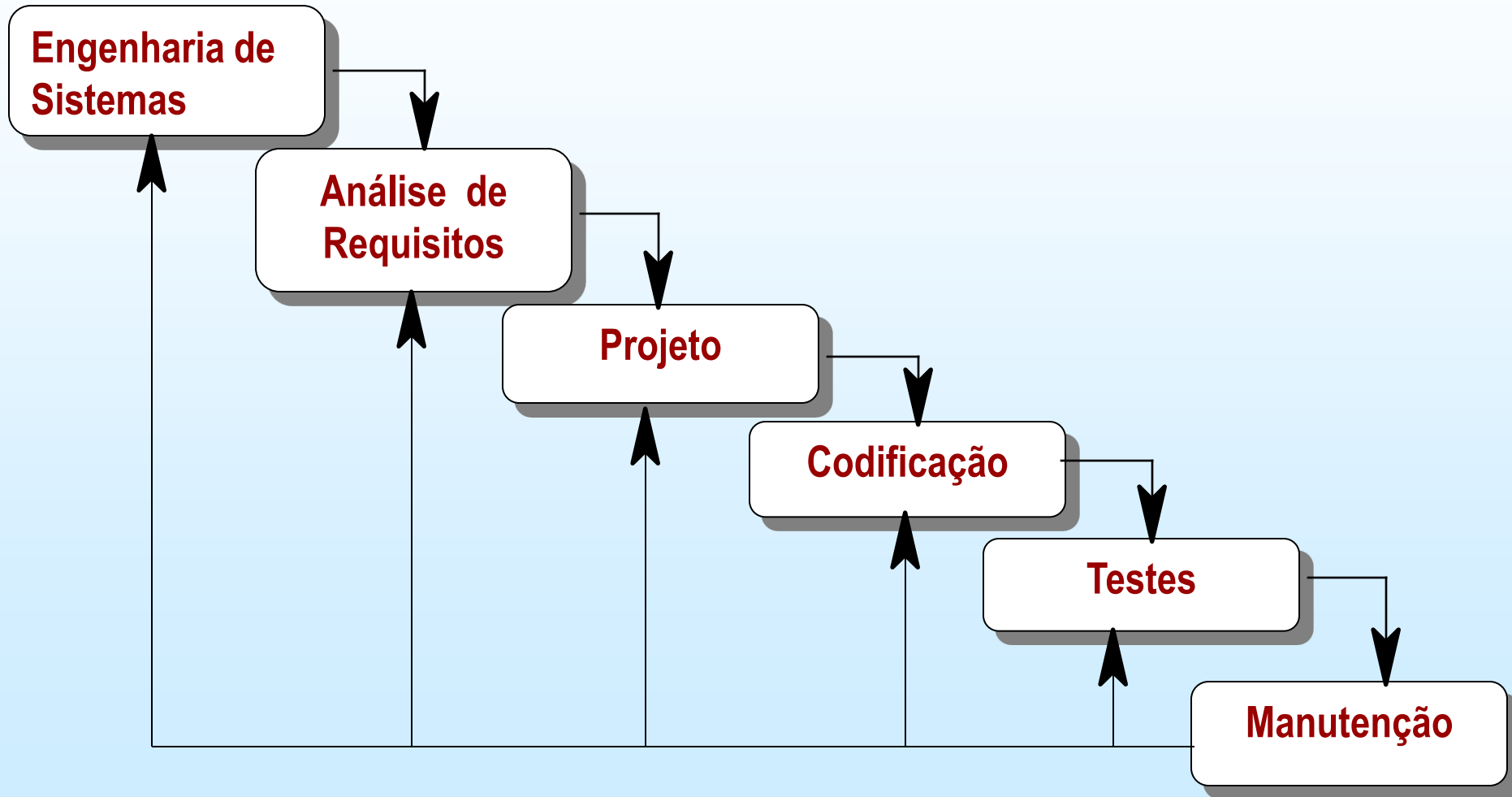
 Modelo Incremental

 Modelo Espiral

Modelo Cascata (1/9)

- ↪ modelo mais antigo e o mais amplamente usado da engenharia de software
- ↪ requer uma abordagem sistemática, sequencial ao desenvolvimento de software
- ↪ enfoque desse modelo nos documentos e nos artefatos

Modelo Cascata (2/9)



Modelo Cascata (3/9)

Engenharia de
Sistemas

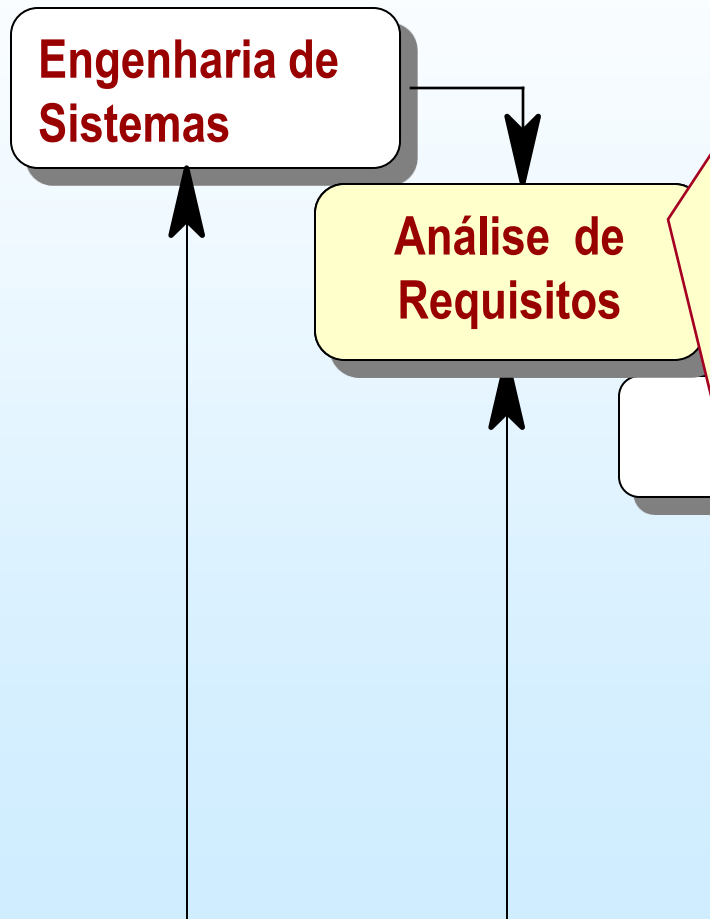
Engenharia de Sistemas

↳ coleta de requisitos em nível de sistema

Testes

Manutenção

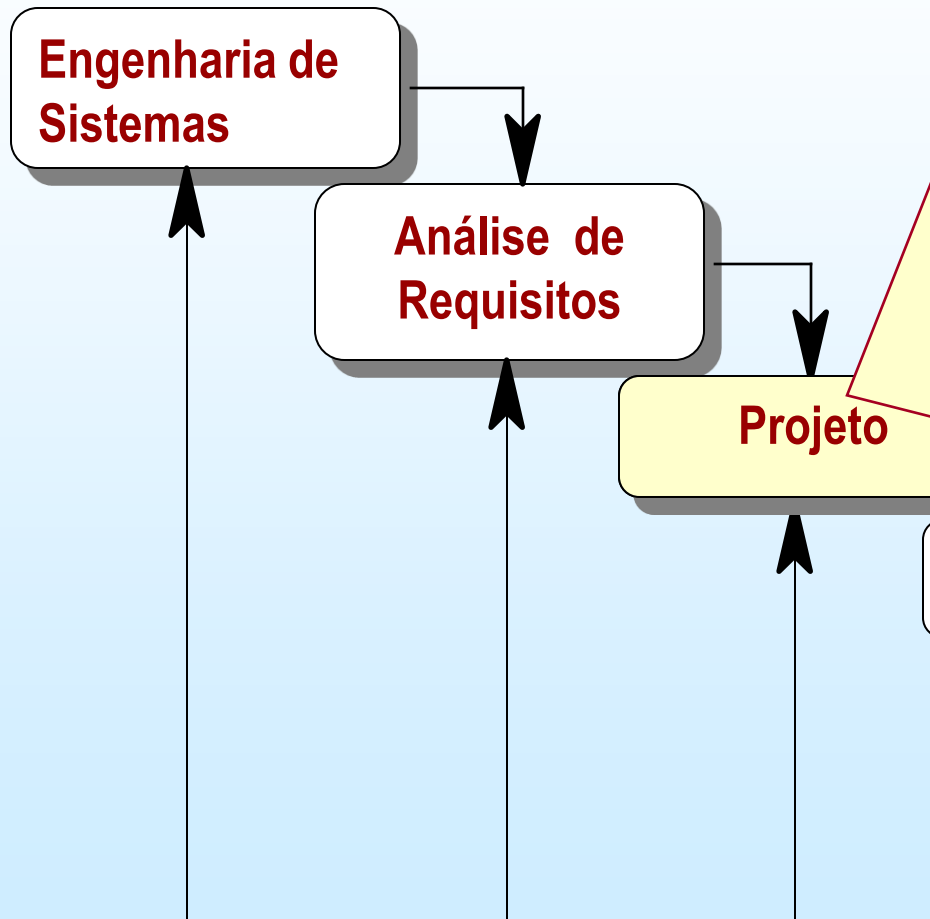
Modelo Cascata (4/9)



Análise de Requisitos de Software

- ↪ coleta dos requisitos concentrado especificamente no software
- ↪ os requisitos (para o sistema e para o software) são documentados e revistos com o cliente

Modelo Cascata (5/9)



Projeto

↳ tradução dos requisitos do software para um conjunto de representações que podem ser avaliadas quanto à qualidade, antes que a codificação se inicie

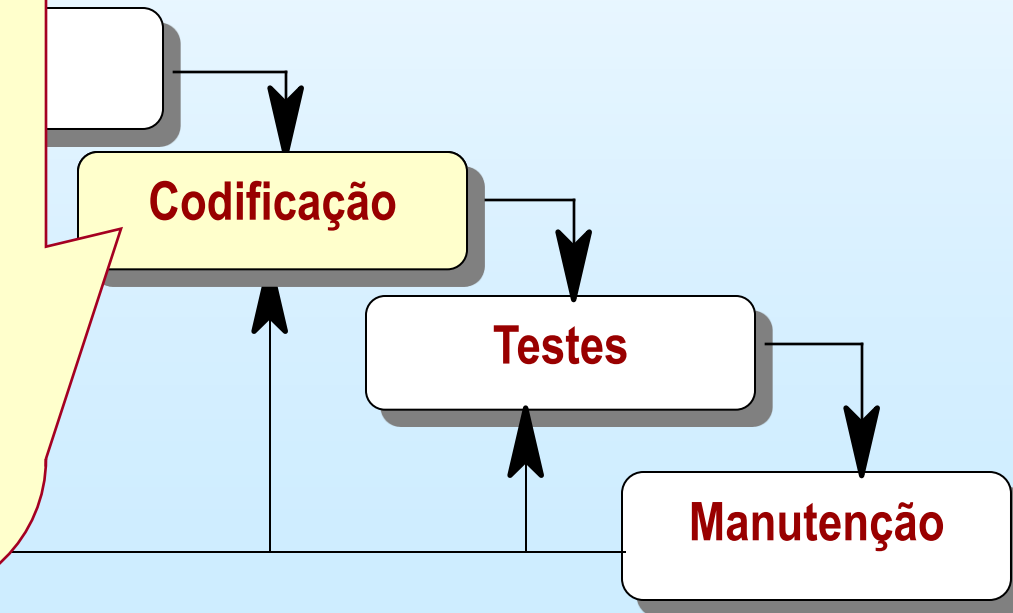
- ↳ *Estrutura de Dados*
- ↳ *Arquitetura de Software*
- ↳ *Detalhes Procedimentais e*
- ↳ *Caracterização de Interfaces*

Modelo Cascata (6/9)

Engenharia de
Sistemas

Codificação

↳ tradução das representações do projeto em instruções executáveis pelo computador

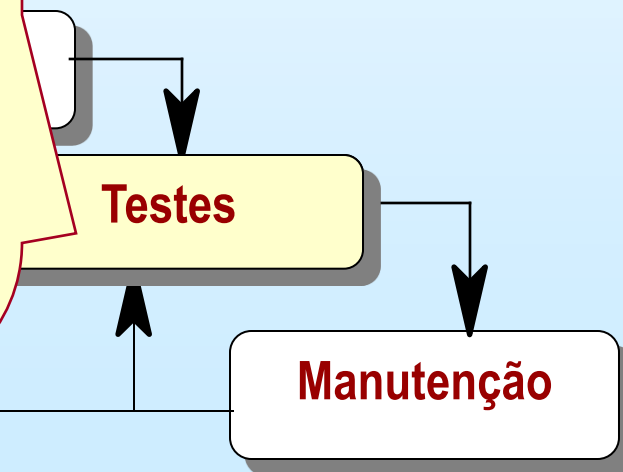


Modelo Cascata (7/9)

Testes

↘ Concentra-se:

- nos aspectos lógicos internos do software ⇒ garantia que todas as instruções tenham sido testadas
- nos aspectos funcionais externos do software ⇒ garantia que a entrada definida produza resultados que concordem com os esperados



Modelo Cascata (8/9)

Manutenção

↪ provavelmente o software deverá sofrer mudanças depois que for entregue ao cliente

↪ causas das mudanças

- erros – *Manut. Corretiva*
- adaptação do software – *Manut. Adaptativa*
- acréscimos funcionais e de desempenho – *Manut. Evolutiva*
- melhoria da manutenibilidade – *Manut. Preditiva*



Manutenção

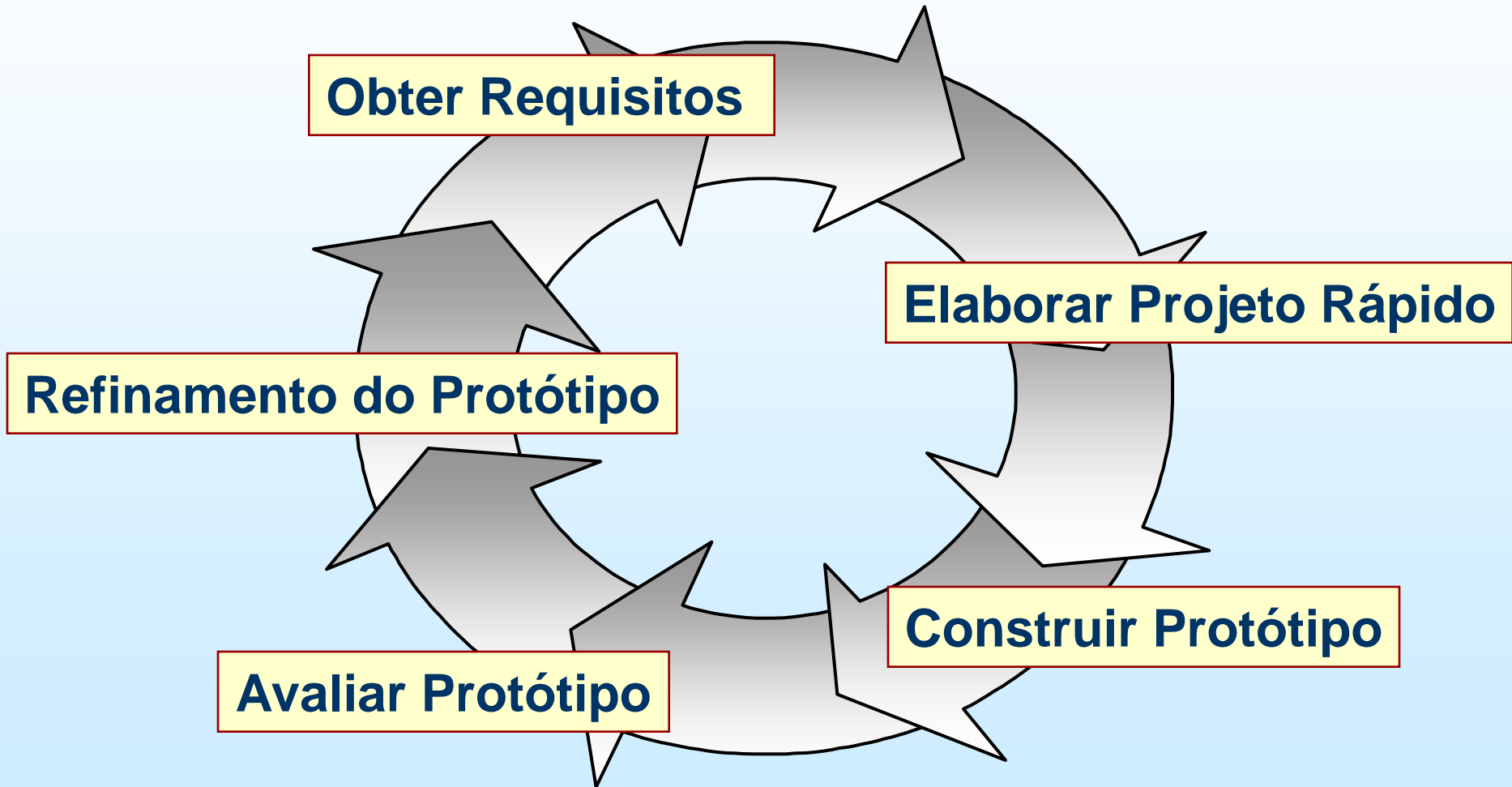
Problemas com o Modelo Cascata (9/9)

- 💣 Projetos reais raramente seguem o fluxo seqüencial que o modelo propõe
- 💣 Logo no início é difícil estabelecer explicitamente todos os requisitos \Rightarrow no começo dos projetos sempre existe uma incerteza natural
- 💣 O cliente deve ter paciência \Rightarrow uma versão executável do software só fica disponível numa etapa avançada do desenvolvimento

Modelo de Prototipação (1/9)

- ↳ Objetivo: entender os requisitos do usuário
- ↳ possibilita que o desenvolvedor crie um modelo (protótipo) do software que deve ser construído
- ↳ apropriado quando o cliente definiu um conjunto de objetivos gerais para o software, mas não identificou detalhadamente esses requisitos

Modelo de Prototipação (2/9)



Modelo de Prototipação (3/9)



Obter Requisitos

Desenvolvedor e cliente:

- ↪ definem os objetivos gerais do software
- ↪ identificam quais requisitos são conhecidos
- ↪ identificam as áreas que necessitam de definições adicionais

Modelo de Prototipação (4/9)



Obter Requisitos

Elaborar Projeto Rápido

Representação dos aspectos do software que são visíveis ao usuário

- ↪ abordagens de entrada e
- ↪ formatos de saída

Modelo de Prototipação (5/9)

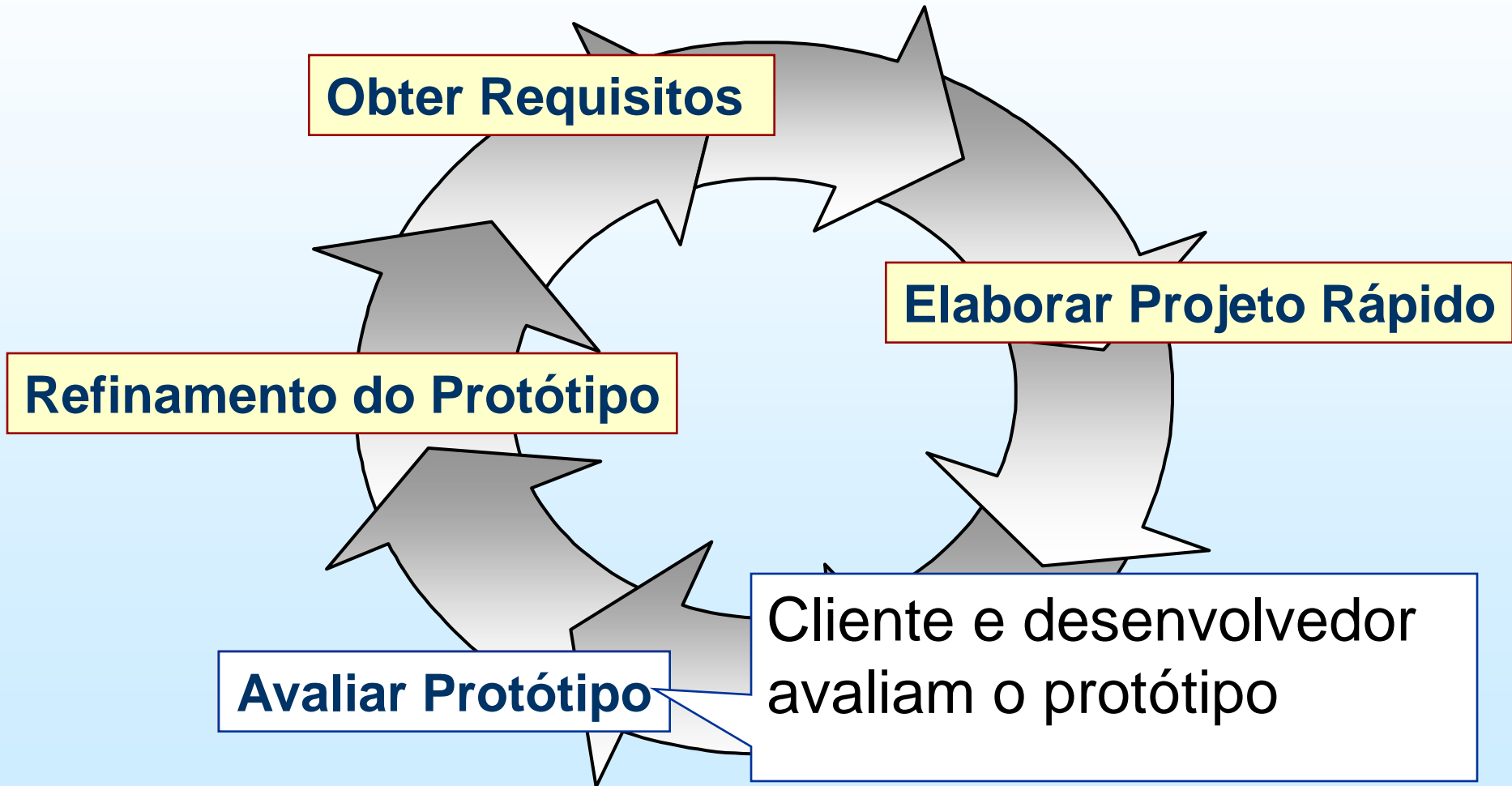
Obter Requisitos

Elaborar Projeto Rápido

Implementação rápida do projeto

Construir Protótipo

Modelo de Prototipação (6/9)



Modelo de Prototipação (7/9)



Obter Requisitos

Elaborar Projeto Rápido

Refinamento do Protótipo

Cliente e desenvolvedor refinam os requisitos do software a ser desenvolvido

Modelo de Prototipação (8/9)

```
graph TD; A[Obter Requisitos] --> B[Elaborar Projeto Rápido]; B --> C[CONSTRUÇÃO DO PRODUTO]; C --> A;
```

Obter Requisitos

**CONSTRUÇÃO DO
PRODUTO**

Elaborar Projeto Rápido

Identificados os requisitos, o protótipo deve ser descartado e a versão de produção deve ser construída considerando os critérios de qualidade

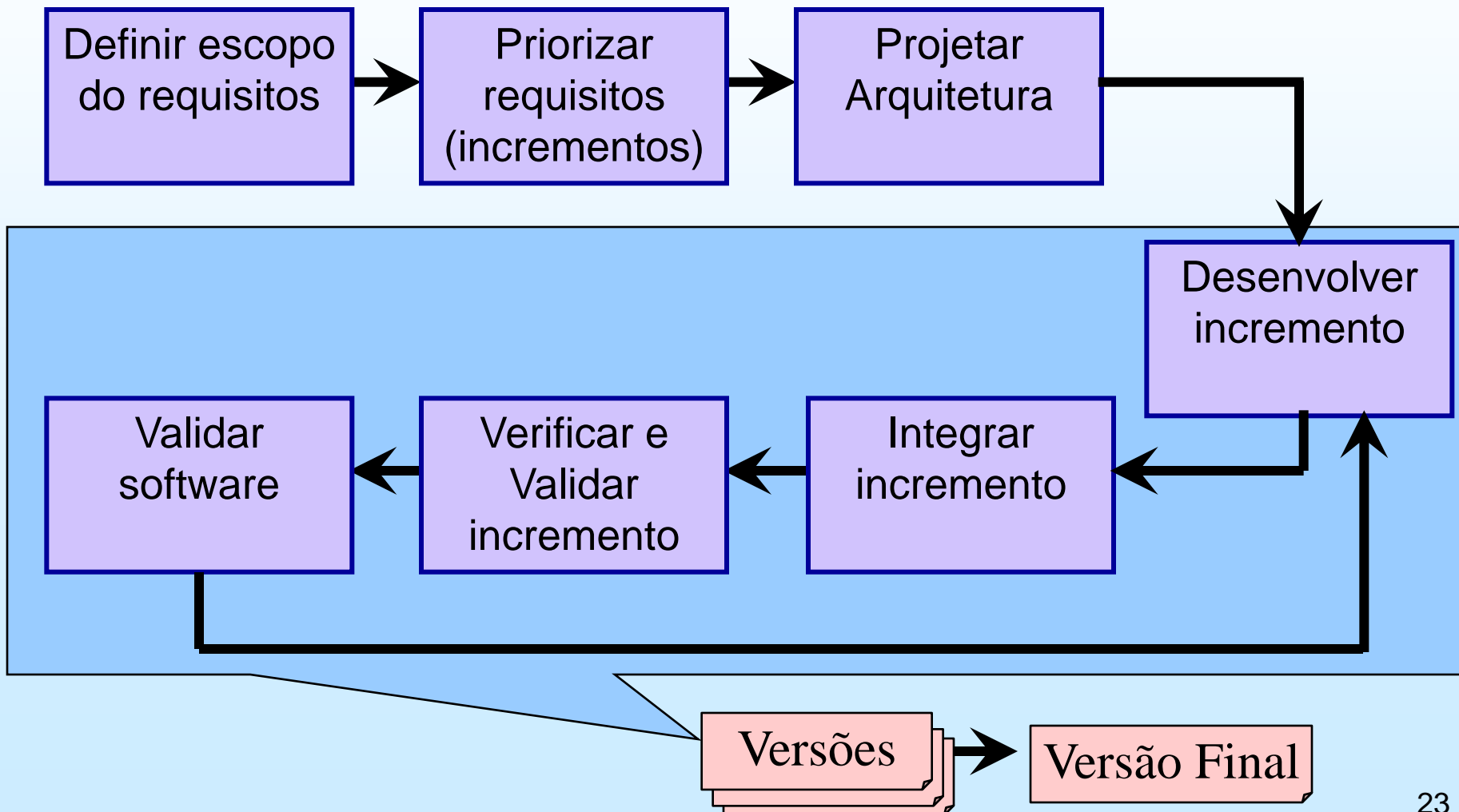
Problemas com a Prototipação (9/9)

- ↪ a qualidade global e a manutenibilidade ao longo do desenvolvimento não foram consideradas
- ↪ utiliza-se o que há disponível na implementação com o objetivo de produzir rapidamente um protótipo
⇒ implementação comprometida
- ↪ Ainda que possam ocorrer problemas, a prototipação é um ciclo de vida eficiente
- ↪ Chave ⇒ definir as regras do jogo logo no começo
 - Acordo entre cliente e desenvolvedor ⇒ o protótipo é construído para servir como um mecanismo a fim de definir os requisitos

Modelo Incremental (1/4)

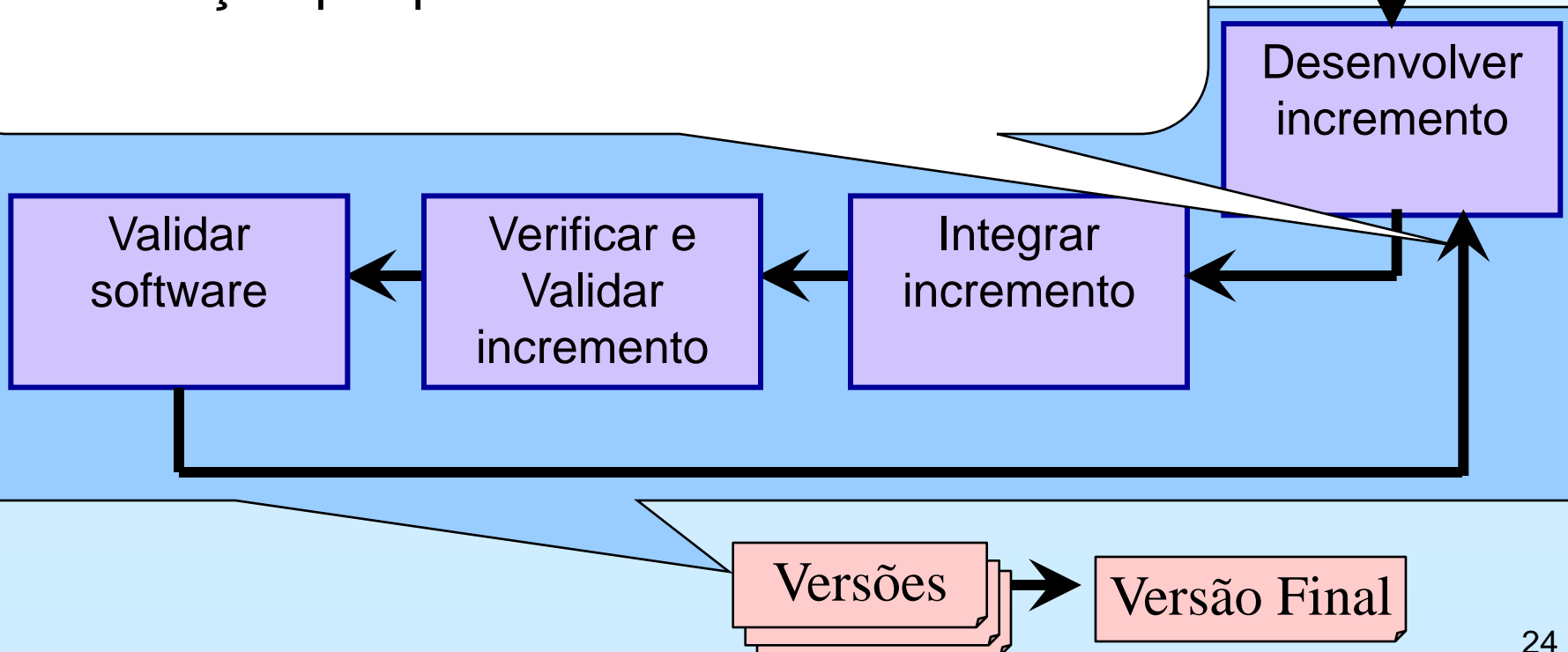
- ↳ Tem os benefícios do modelo cascata e da prototipação
- ↳ Ideia básica: desenvolvimento (projeto, codificação e teste) incremental do software
- ↳ Priorização das funcionalidades principais do software

Modelo Incremental (2/4)



Modelo Incremental (3/4)

Um **plano** é desenvolvido para o próximo incremento, como resultado do uso e/ou avaliação por parte do cliente



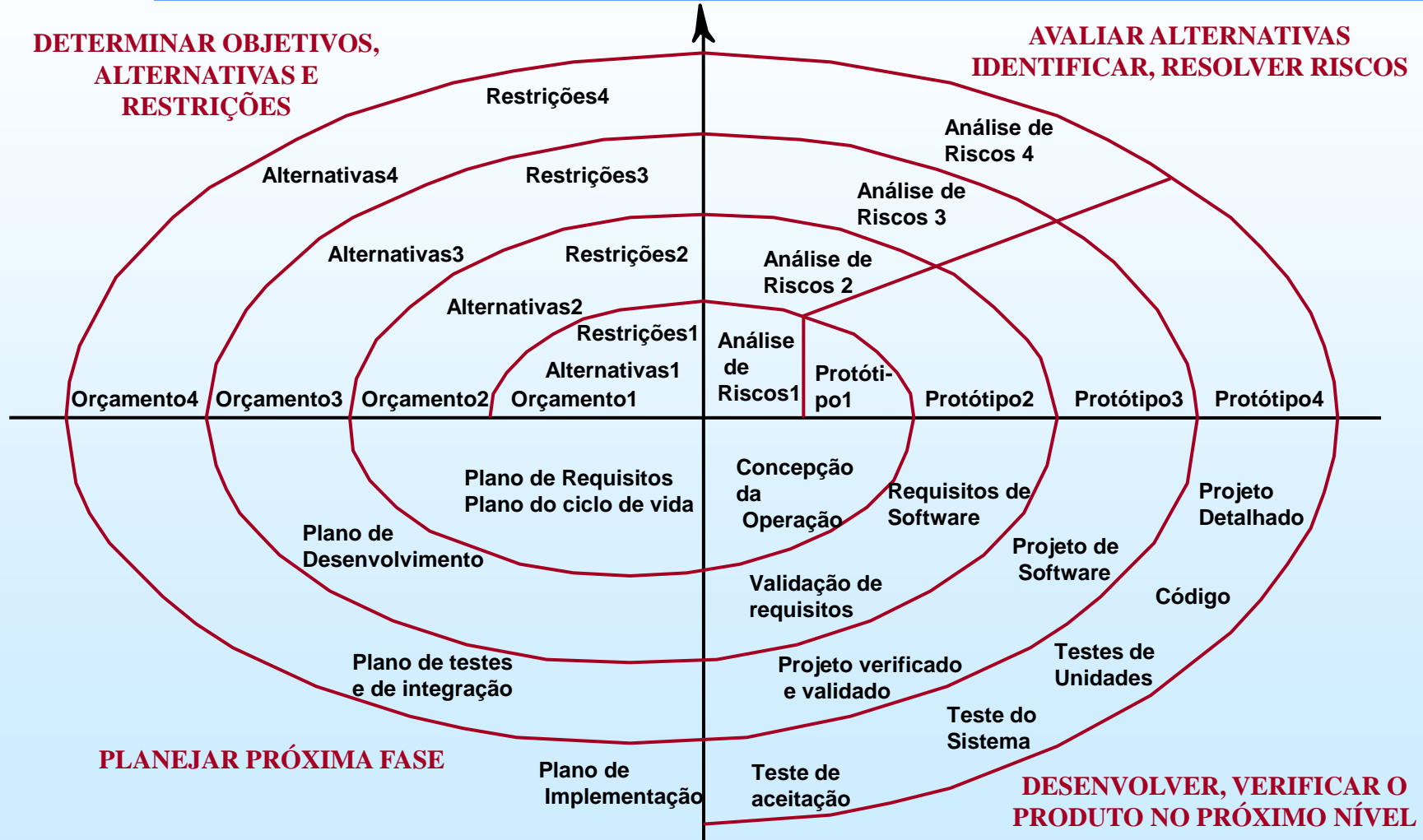
Modelo Incremental (4/4)

- ↪ Os clientes não precisam esperar que a versão final do software seja entregue para usá-lo
- ↪ Os clientes podem utilizar os primeiros incrementos desenvolvidos como um protótipo, de forma a definir melhor alguns requisitos do software
- ↪ Existe um risco menor de fracasso do software
- ↪ Como as funções prioritárias são entregues primeiro, é inevitável que estas passem por um período de testes mais intensivo

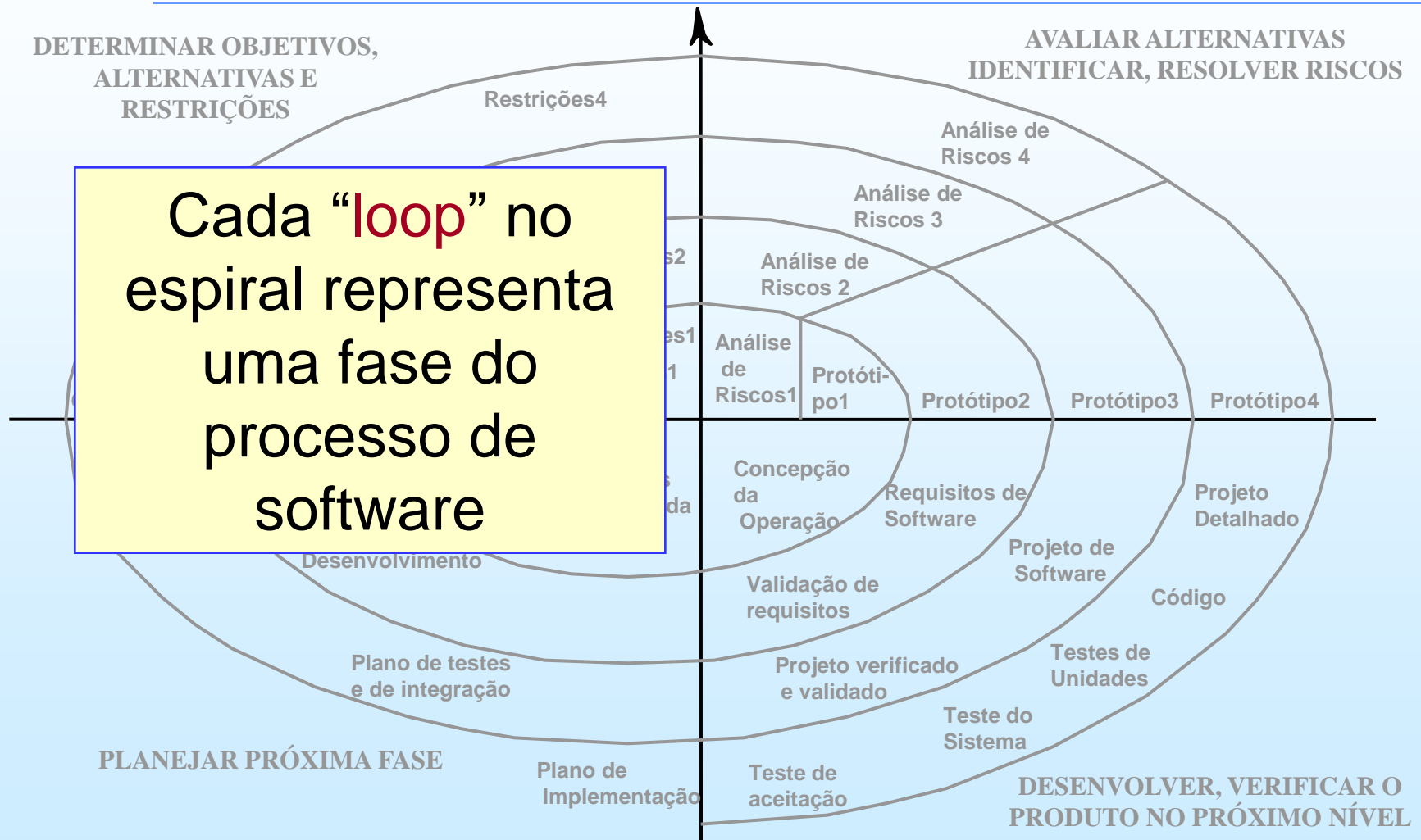
Modelo Espiral (1/11)

- ↪ Engloba as melhores características do modelo cascata e da prototipação, adicionando um novo elemento: a Análise de Risco
- ↪ Usa a prototipação, em qualquer etapa da evolução do produto, como mecanismo de redução de riscos
- ↪ 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

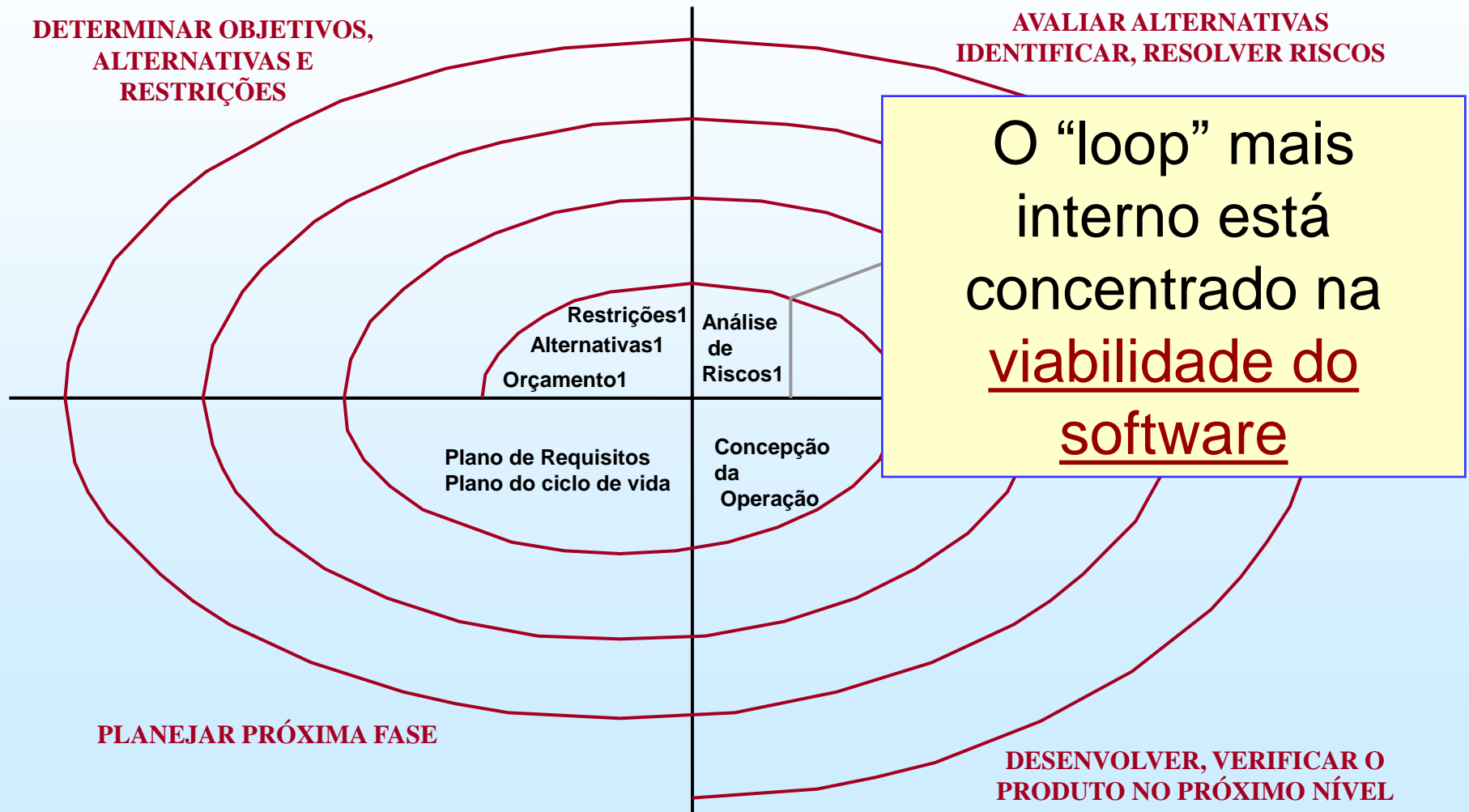
Modelo Espiral (com 4 regiões) (2/11)



Modelo Espiral (com 4 regiões) (3/11)



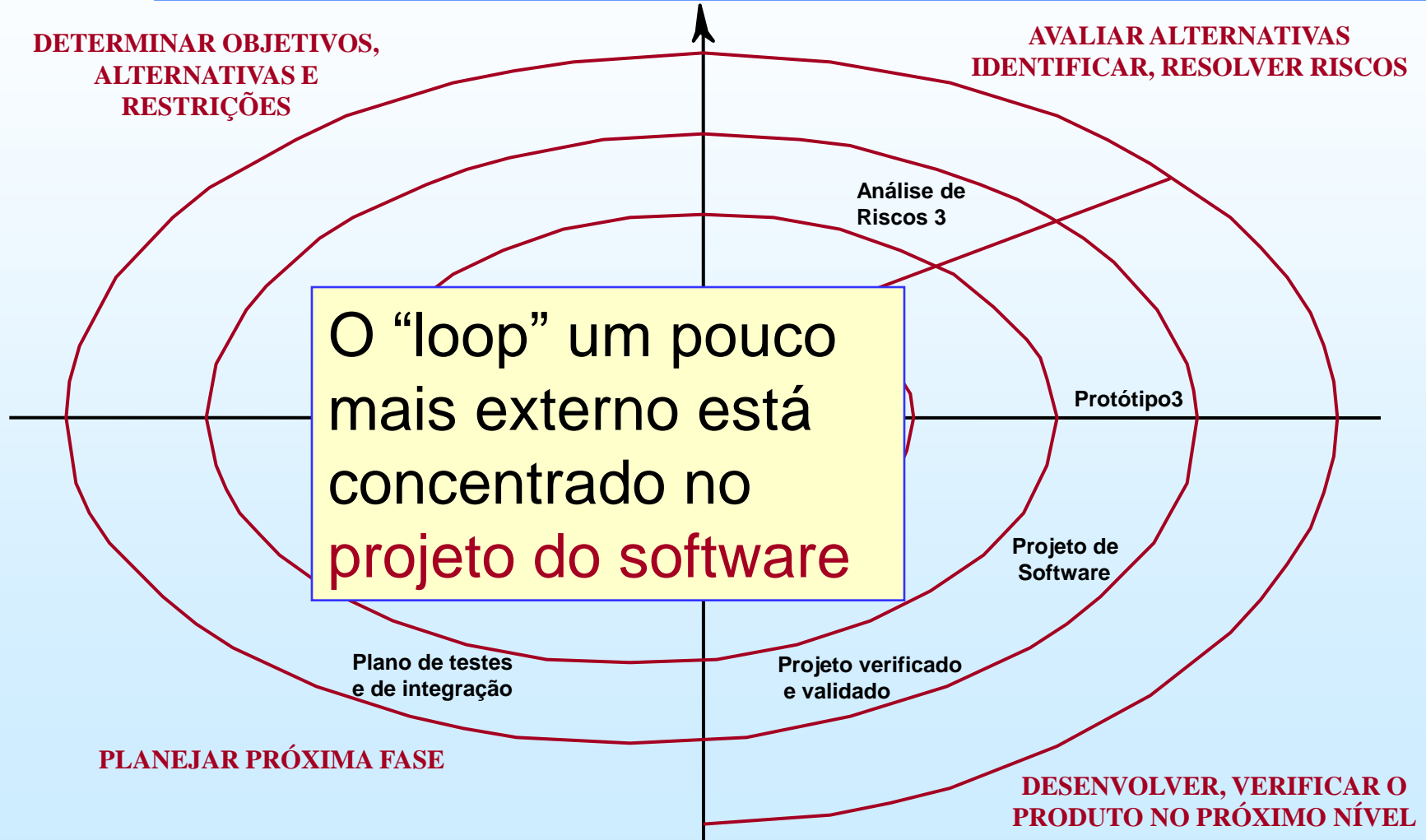
Modelo Espiral (com 4 regiões) (4/11)



Modelo Espiral (com 4 regiões) (5/11)



Modelo Espiral (com 4 regiões) (6/11)



Modelo Espiral (com 4 regiões) (7/11)



Modelo Espiral (com 4 regiões) (8/11)

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

Modelo Espiral (com 6 regiões) (10/11)



Comentários sobre o Ciclo de Vida em Espiral (11/11)

- ↪ Exige considerável experiência na determinação de riscos e depende dessa experiência para ter sucesso
- ↪ Possibilita ao desenvolvedor e ao cliente entender e reagir aos riscos em cada etapa evolutiva






Modelos de Processo de Software

✚ Modelos de Processos Prescritivos

- ✚ Visam assegurar a aderência às normas e padrões estabelecidos
- ✚ Processos com forte ênfase em controles e documentações
- ✚ Voltados para equipes grandes e/ou dispersas
- ✚ Em geral, especificam várias atividades além do desenvolvimento de software
- ✚ Buscam satisfazer o maior número possível de contextos, para serem usados como o único processo
- ✚ Atendem a muitas exigências do CMMI e ISO
- ✚ Geralmente são “personalizados” para cada projeto e/ou empresa

Modelos de Processo de Software

Alguns Modelos de Processos Prescritivos

-  Unified Process (UP/RUP) e suas instanciações (EUP – *Enterprise Unified Process*)
-  MSF (*Microsoft Solutions Framework*)
-  ICONIX
-  CATALISYS
-  OPEN (*Object-oriented Process, Environment and Notation*)

UP (1/7)

↪ UP é RUP sem o *copyrighted* de produto Rational Unified Process

O UP baseia-se em três princípios:

↪ É direcionado por Use Cases

- Cada uma das interações dos diferentes tipos de usuário (atores) com o sistema é chamada **Use Case**

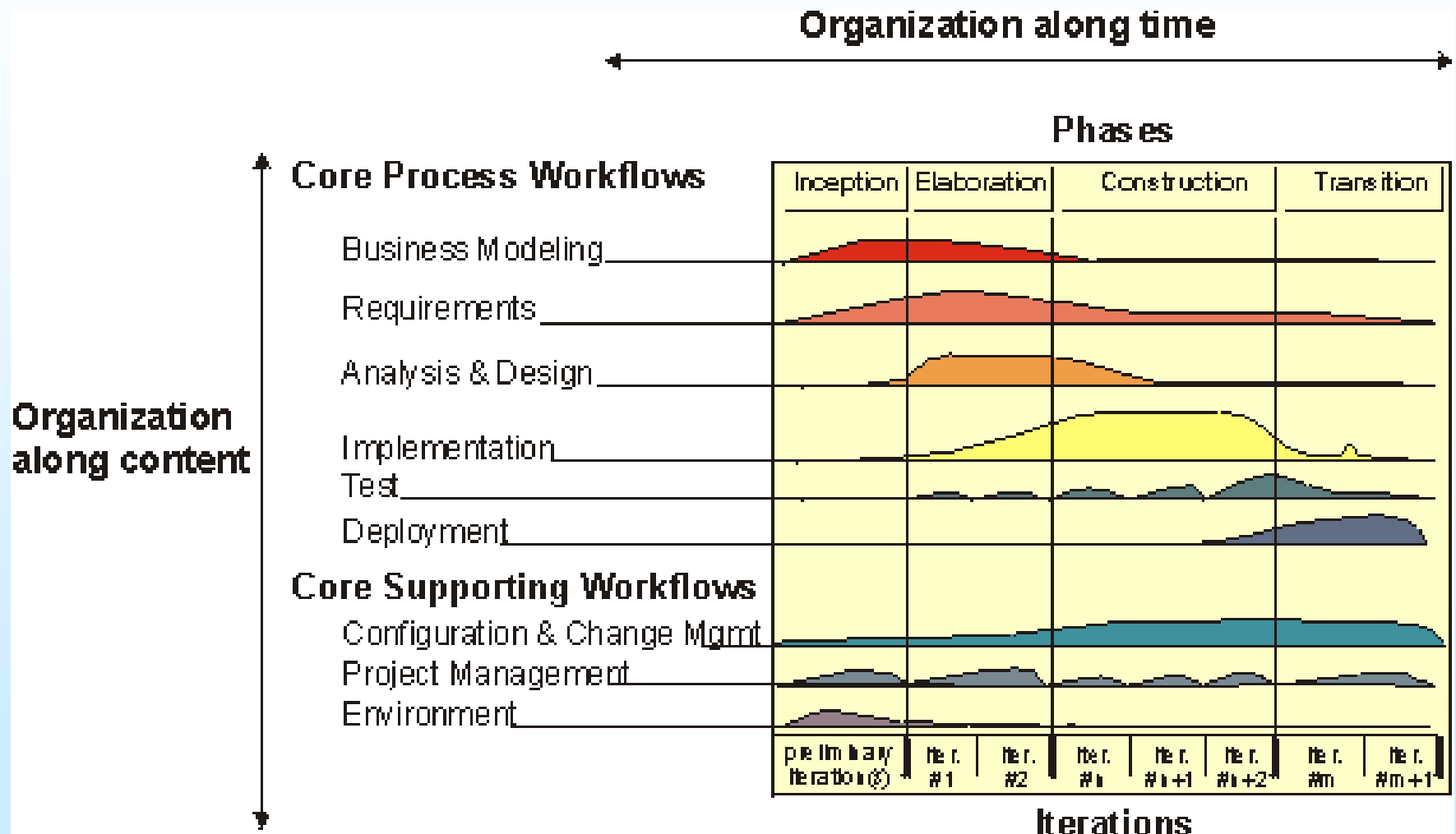
↪ É centrado em Arquitetura

- A **Arquitetura** de um sistema refere-se às diferentes visões que compõem a solução completa (abstrações)

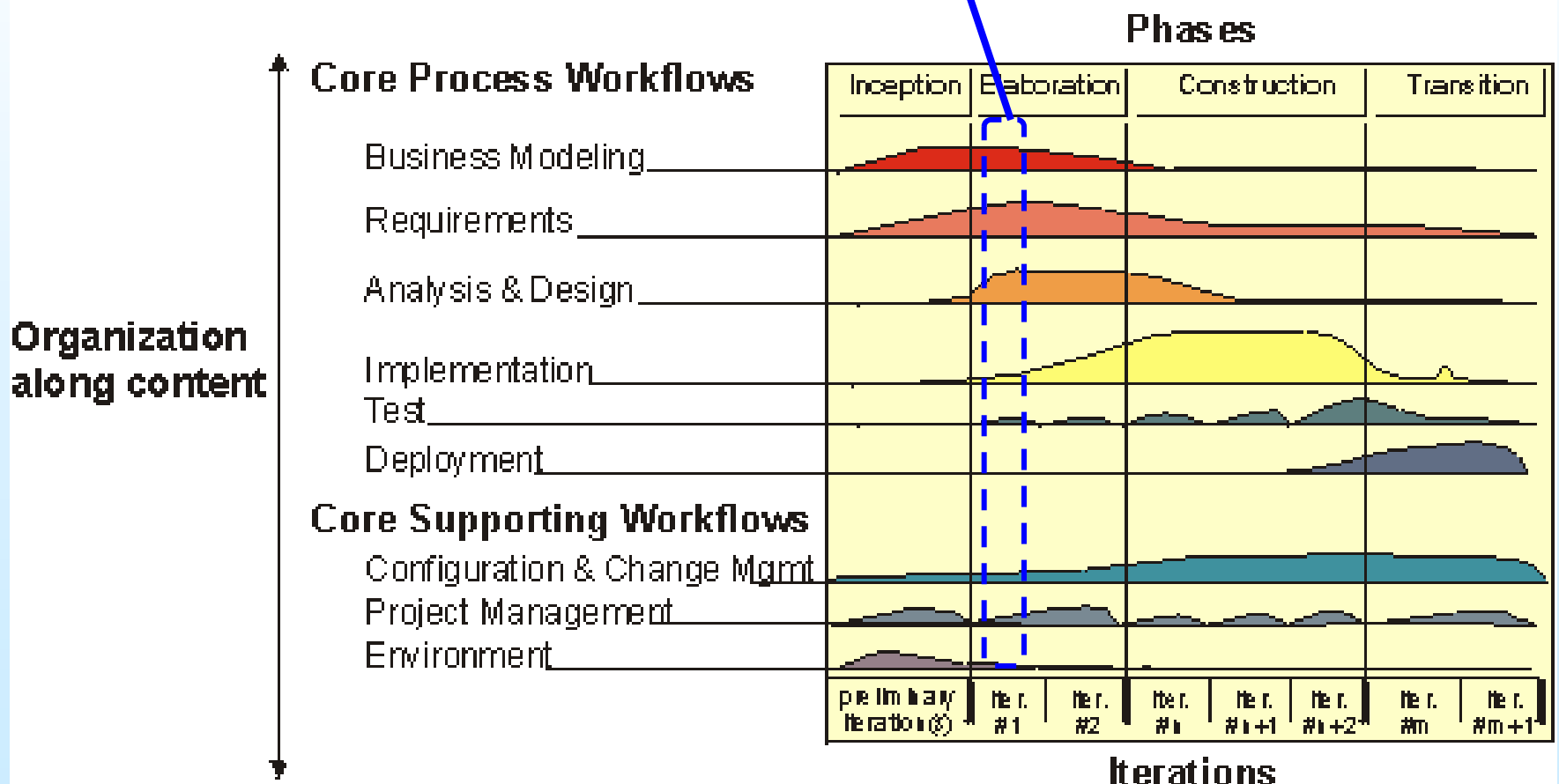
↪ É Iterativo e Incremental

- Projeto para desenvolvimento de sistema deve ser quebrado em “mini-projetos”, chamados **Iterações**

UP (2/7)



- Uma iteração é uma seqüência completa de disciplinas (*workflow*).
- O resultado de uma iteração é um artefato ou uma série de artefatos.
- Um artefato pode ser um código-fonte, um modelo, etc.
- Uma vez que os artefatos para uma fase específica foram completados ocorre a passagem para a próxima fase.

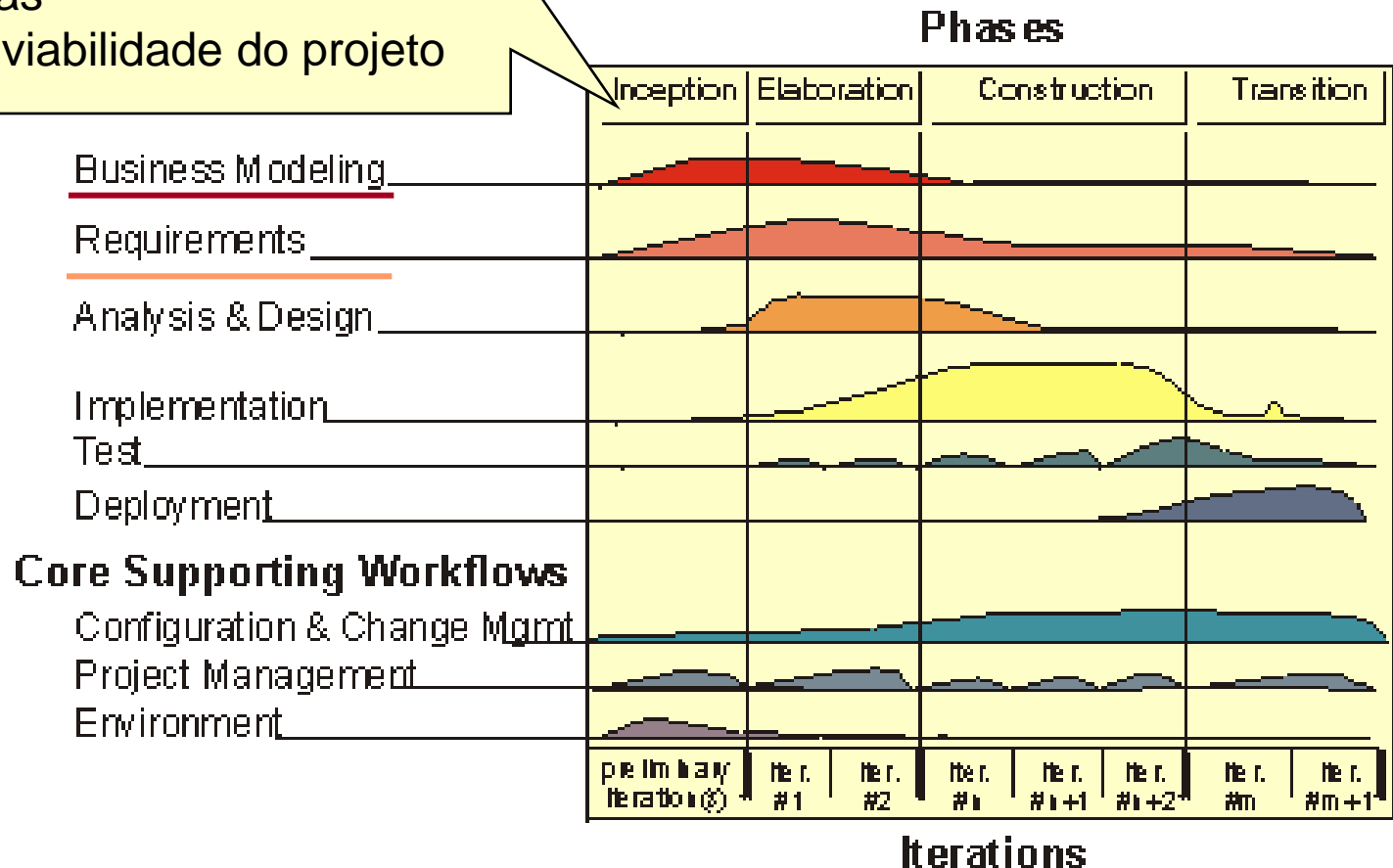


LIP (4/7)

- Define-se o escopo do projeto
 - Avalia-se a tecnologia
 - Relaciona-se os riscos principais
 - Detecta-se áreas mais críticas a serem tratadas
- Verifica-se a viabilidade do projeto

Organization along time

Organization along content



UP (5/7)

Especifica-se os requisitos em detalhes
Identifica-se e avalia-se a arquitetura do software

Organization along time

Phases

Core Process Workflows

Business Modeling

Requirements

Analysis & Design

Implementation

Test

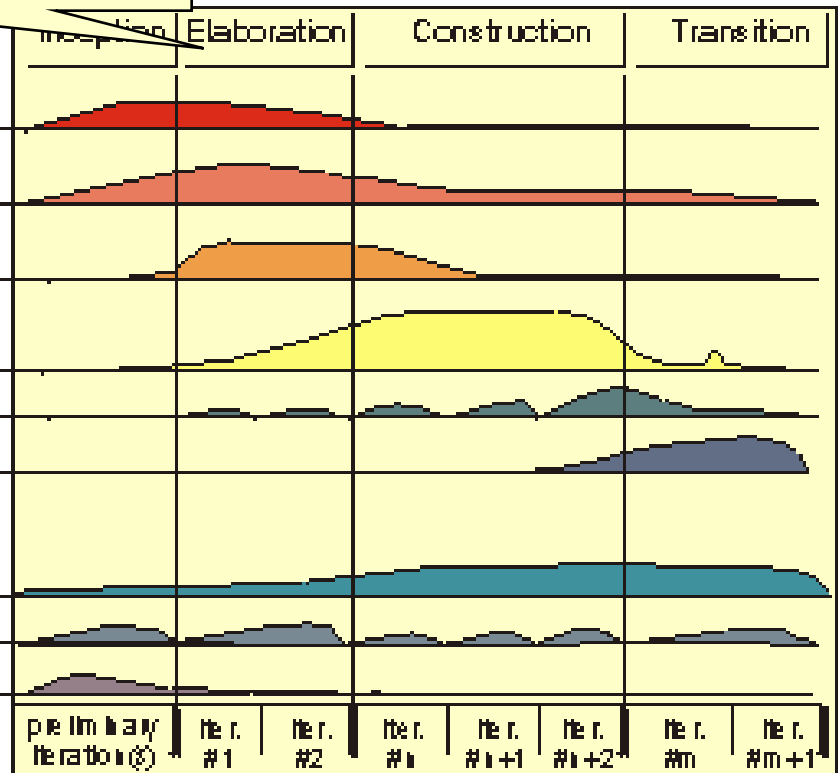
Deployment

Core Supporting Workflows

Configuration & Change Mgmt

Project Management

Environment



Iterations

Organization
along content

UP (6/7)

- Constrói-se o produto
- Testa-se as versões beta do produto

Organization
along content

Core Process workflows

Business Modeling

Requirements

Analysis & Design

Implementation

Test

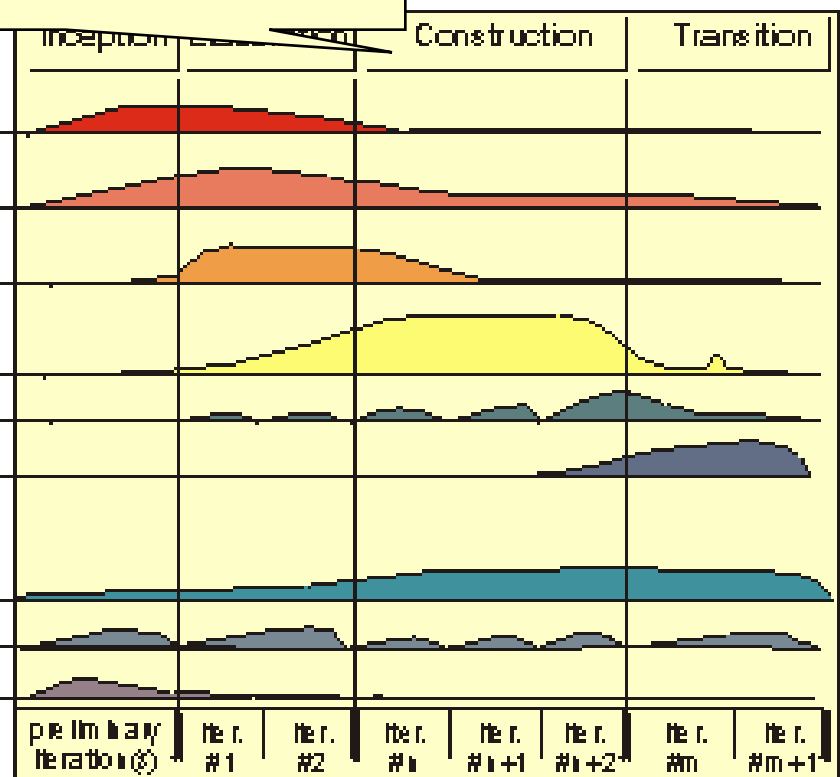
Deployment

Core Supporting Workflows

Configuration & Change Mgmt

Project Management

Environment



Iterations

UP (8/7)

Testa-se a versão do produto para entrega ao cliente
Implanta-se o sistema

Organization
along content

Core Process Workflows

Business Modeling

Requirements

Analysis & Design

Implementation

Test

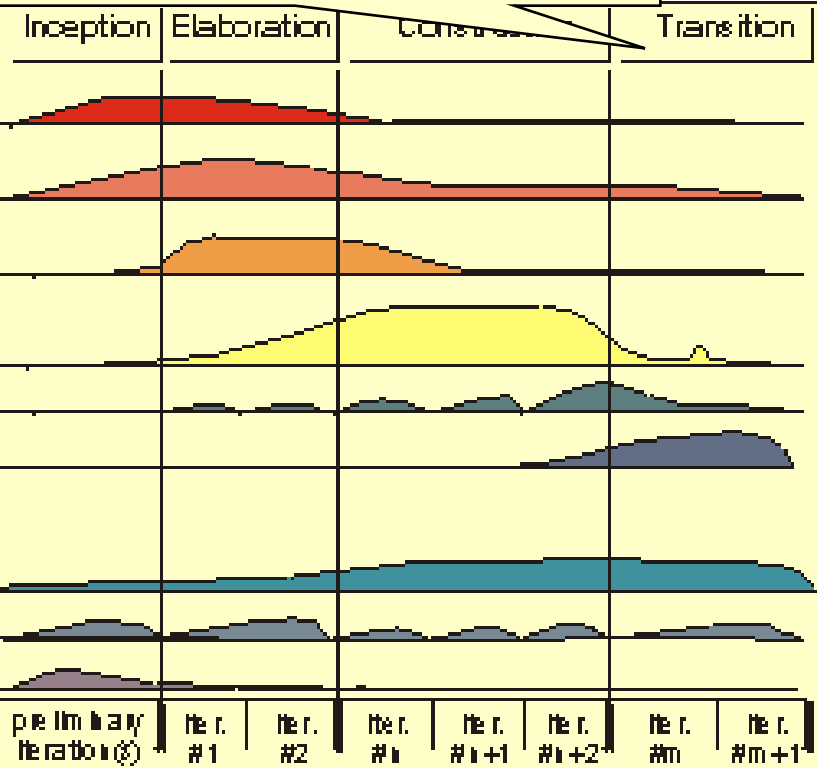
Deployment

Core Supporting Workflows

Configuration & Change Mgmt

Project Management

Environment



Iterations

EUP – Enterprise Unified Process (1/2)

↪ Extensão do RUP

↪ Essa extensão inclui:

- Duas novas fases:
 - Produção: apoio na operação e suporte
 - Retirada: retirar o sistema em operação. A troca de sistema legados por novos sistemas deve ter impacto mínimo nas operações de negócio
- Novas disciplinas:
 - Modelagem do Negócio da Empresa, Gestão de Portfólio, Arquitetura Empresarial, Reuso Estratégico, Gestão de Pessoas, Administração Empresarial, Melhoria do Processo de Software

EUP (2/2)

Development Disciplines

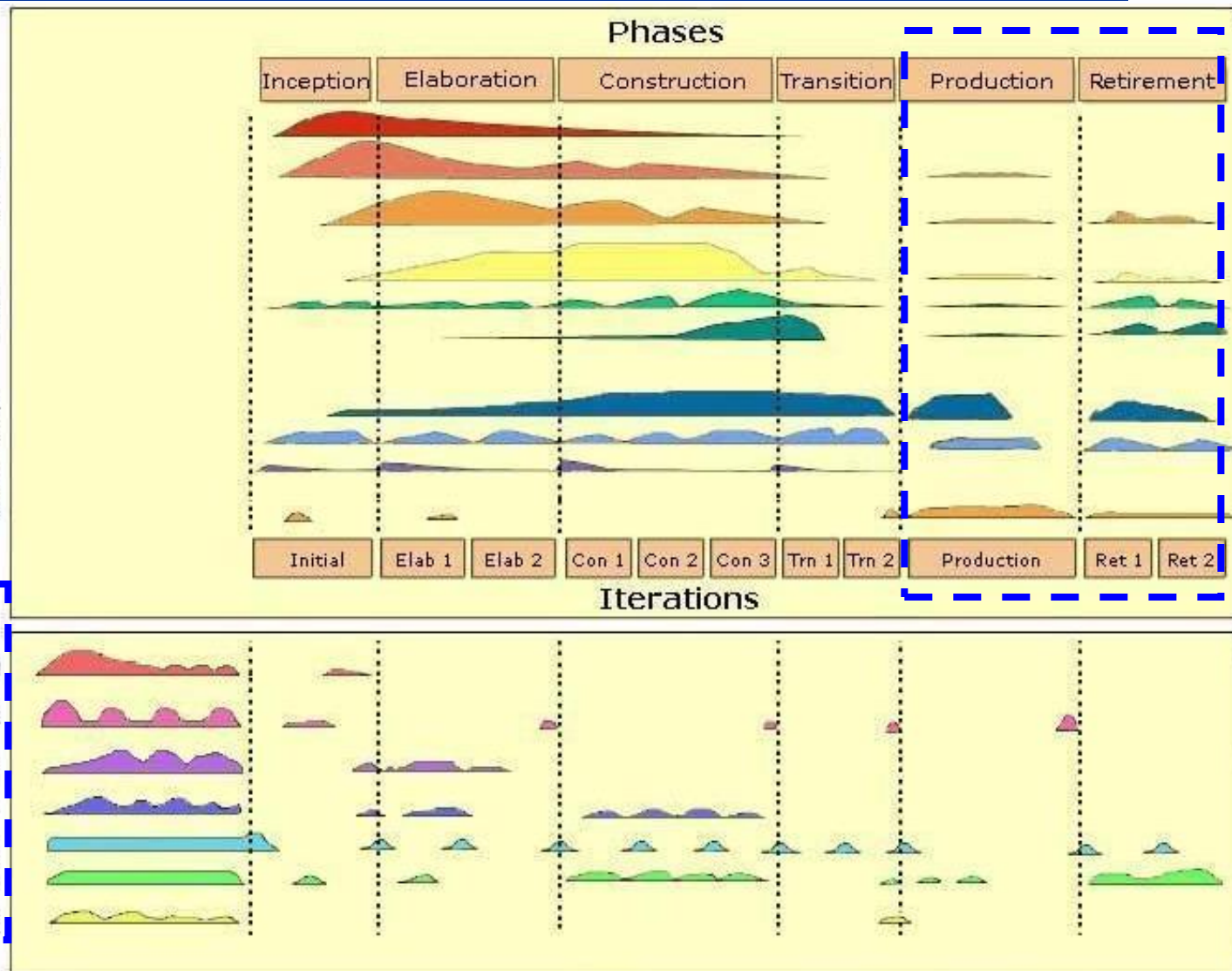
Business Modeling
Requirements
Analysis & Design
Implementation
Test
Deployment

Support Disciplines

Configuration and Change Mgmt.
Project Management
Environment
Operations & Support

Enterprise Disciplines

Enterprise Business Modeling
Portfolio Management
Enterprise Architecture
Strategic Reuse
People Management
Enterprise Administration
Software Process Improvement



Bibliografia

- ↗ SANCHES, ROSELY. **Material Didático: Engenharia de Software**. ICMC-USP, 2005.
- ↗ PRESSMAN, ROGER S. **Engenharia de Software**. 5ª edição. Rio de Janeiro: McGraw-Hill, 2002.
- ↗ SOMERVILLE, IAN. **Engenharia de Software**. 6ª edição. São Paulo: Addison Wesley, 2003.
- ↗ X-tier SAE Inc. **Using RUP/UP: 10 Easy Steps**. Disponível em: <www.x-tier.com/public/RUPUPIn10EasySteps.doc>. Acessado em: 07 abr 2006.