Modelos de Processo de Software

Engenharia de Software

Profa. Dra. Rosana T. Vaccare Braga

1º semestre de 2017

(material produzido e atualizado pelos professores do grupo de pesquisa em Engenharia de Software do ICMC-USP)

ENGENHARIA DE SOFTWARE

pode ser vista como uma abordagem de desenvolvimento de software elaborada com disciplina e métodos bem definidos.

.....“a construção por múltiplas pessoas de um software de múltiplas versões”

[Parnas 1987]

Processo de software

Conjunto de atividades relacionadas que levam à produção de um produto de software

(Sommerville, 2013)

Incluem:

1. Produtos gerados em cada atividade
2. Papéis envolvidos
3. Pré e pós-condições das atividades

Motivação

- **Processos de software são complexos e dependem de vários fatores como:**
  - **Produto a ser desenvolvido**
  - **Equipe**
  - **Recursos disponíveis**
- **Não existe um processo ideal, ele pode ser adaptado de acordo com o contexto, podendo ser mais formal ou mais flexível.**

Motivação

- **Envolvem criatividade e fatores humanos**
- **Decisões precisam ser tomadas ao longo do processo**
- **Decisões erradas podem levar a prejuizos e perda de oportunidades**

Como obter uma representação simplificada de um processo de software?

Modelos de Processo de Software

Modelos de Processo de Software

- Existem vários *modelos de processo de software* (ou *paradigmas de engenharia de software*)
- Cada um representa uma tentativa de colocar ordem em uma atividade inerentemente caótica

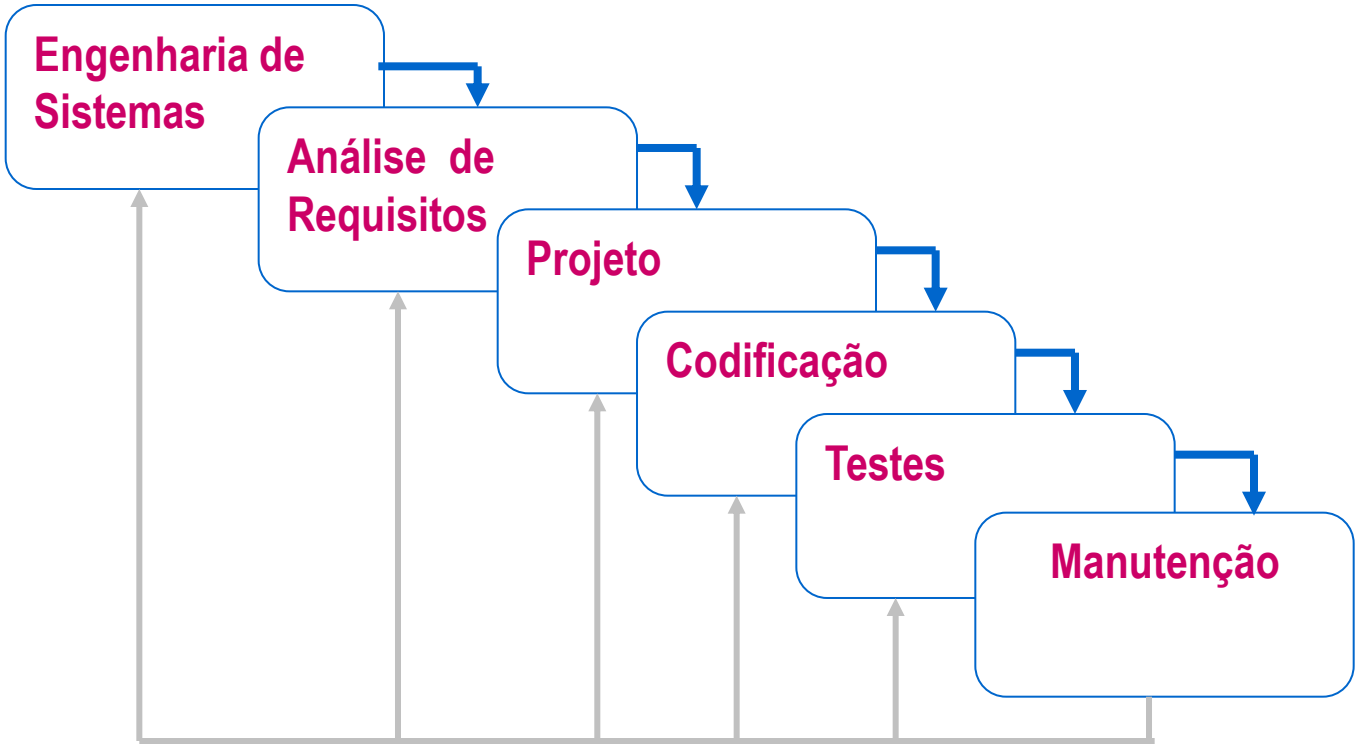
Modelos de Processo de Software

- *O Modelo Sequencial Linear*
  - *também chamado Modelo Cascata*
- *O Modelo de Prototipação*
- *O Modelo RAD (Rapid Application Development)*
- Modelos Evolutivos de Processo de Software
  - O *Modelo Incremental*
    - *O Modelo Ágil*
    - *O Processo Unificado*
  - O *Modelo Espiral*
  - O *Modelo de Montagem de Componentes*
  - O *Modelo de Desenvolvimento Concorrente*
- *Modelo de Métodos Formais*
- *Técnicas de Quarta Geração*

O Modelo Cascata

- modelo mais antigo e o mais amplamente usado da engenharia de software
- modelado em função do ciclo da engenharia convencional
- requer uma abordagem sistemática, seqüencial ao desenvolvimento de software
- o resultado de uma fase se constitui na entrada da outra

O Modelo Cascata



O Modelo Cascata

Engenharia de Sistemas / Informação e Modelagem

- envolve a coleta de requisitos em nível do sistema, com uma pequena quantidade de projeto e análise de alto nível
- esta visão é essencial quando o software deve fazer interface com outros elementos (hardware, pessoas e banco de dados)

O Modelo Cascata

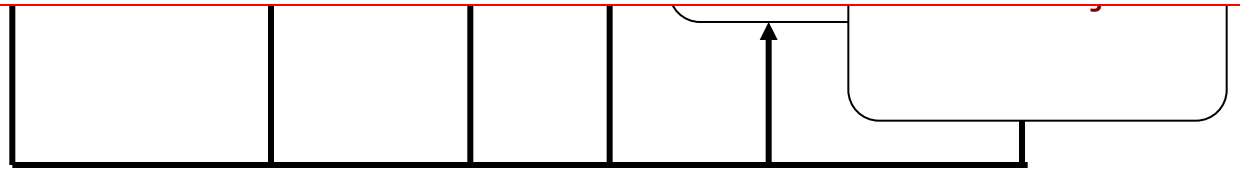
Análise de Requisitos de Software

- o processo de coleta dos requisitos é intensificado e concentrado especificamente no software
- deve-se compreender o domínio da informação, a função, desempenho e interfaces exigidos
- os requisitos (para o sistema e para o software) são documentados e revistos com o cliente

O Modelo Cascata

Projeto

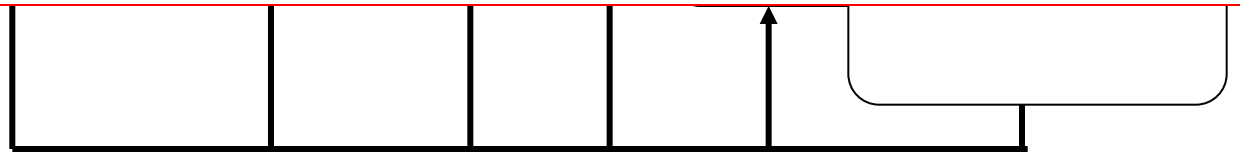
- tradução dos requisitos do software para um conjunto de representações que podem ser avaliadas quanto à qualidade, antes que a codificação se inicie



O Modelo Cascata

Codificação

- tradução das representações do projeto para uma linguagem “artificial” resultando em instruções executáveis pelo computador



O Modelo Cascata

Testes

- Concentra-se:
  - nos aspectos lógicos internos do software, garantindo que todas as instruções tenham sido testadas
  - nos aspectos funcionais externos, para descobrir erros e garantir que a entrada definida produza resultados que concordem com os esperados.

O Modelo Cascata

Manutenção

- provavelmente o software deverá sofrer mudanças depois que for entregue ao cliente
- causas das mudanças: e*rros*, *adaptação do software para acomodar mudanças em seu ambiente externo* e *exigência do cliente para acréscimos funcionais e de desempenho*

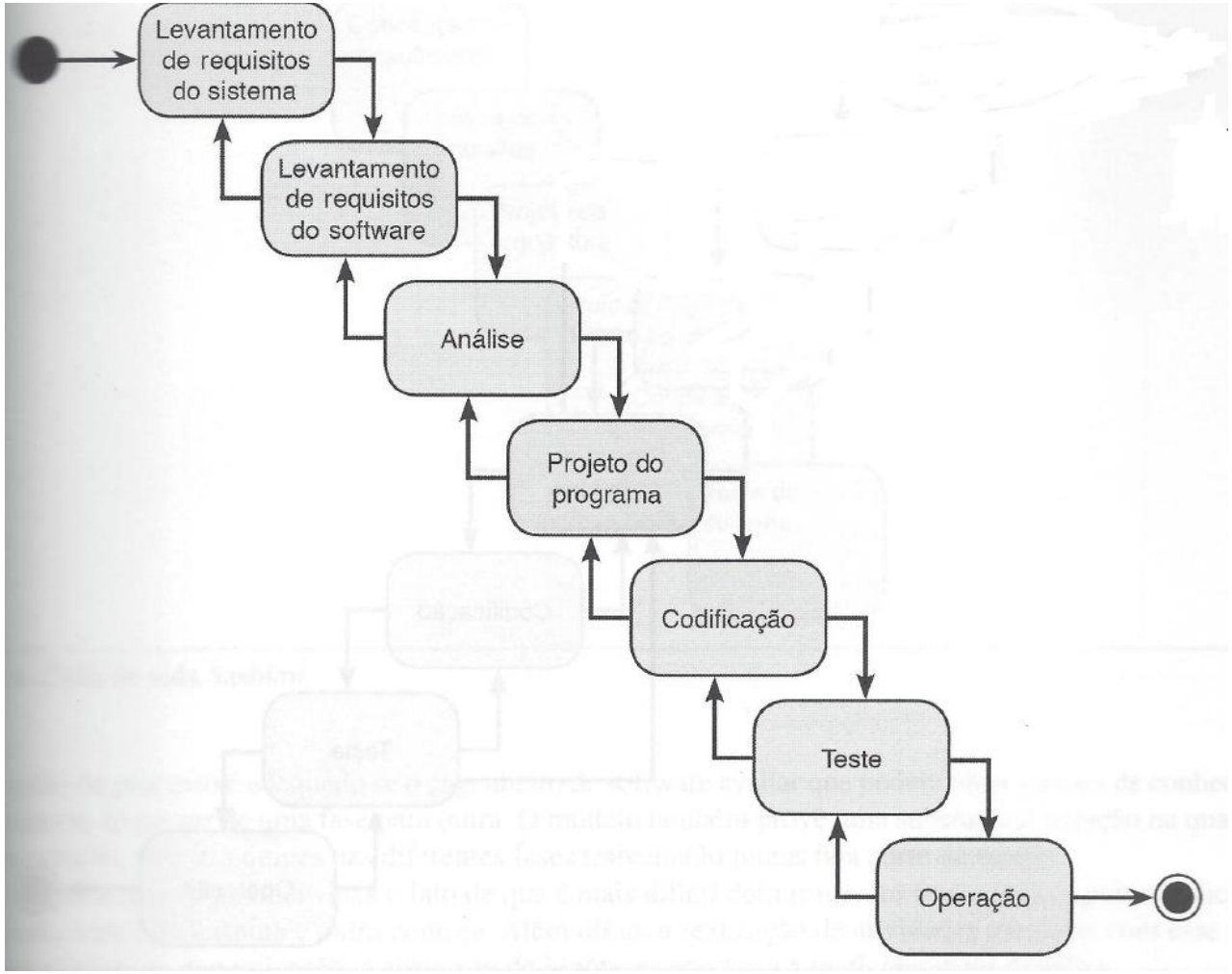
Problemas com o Modelo Cascata

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

Problemas com o Modelo Cascata

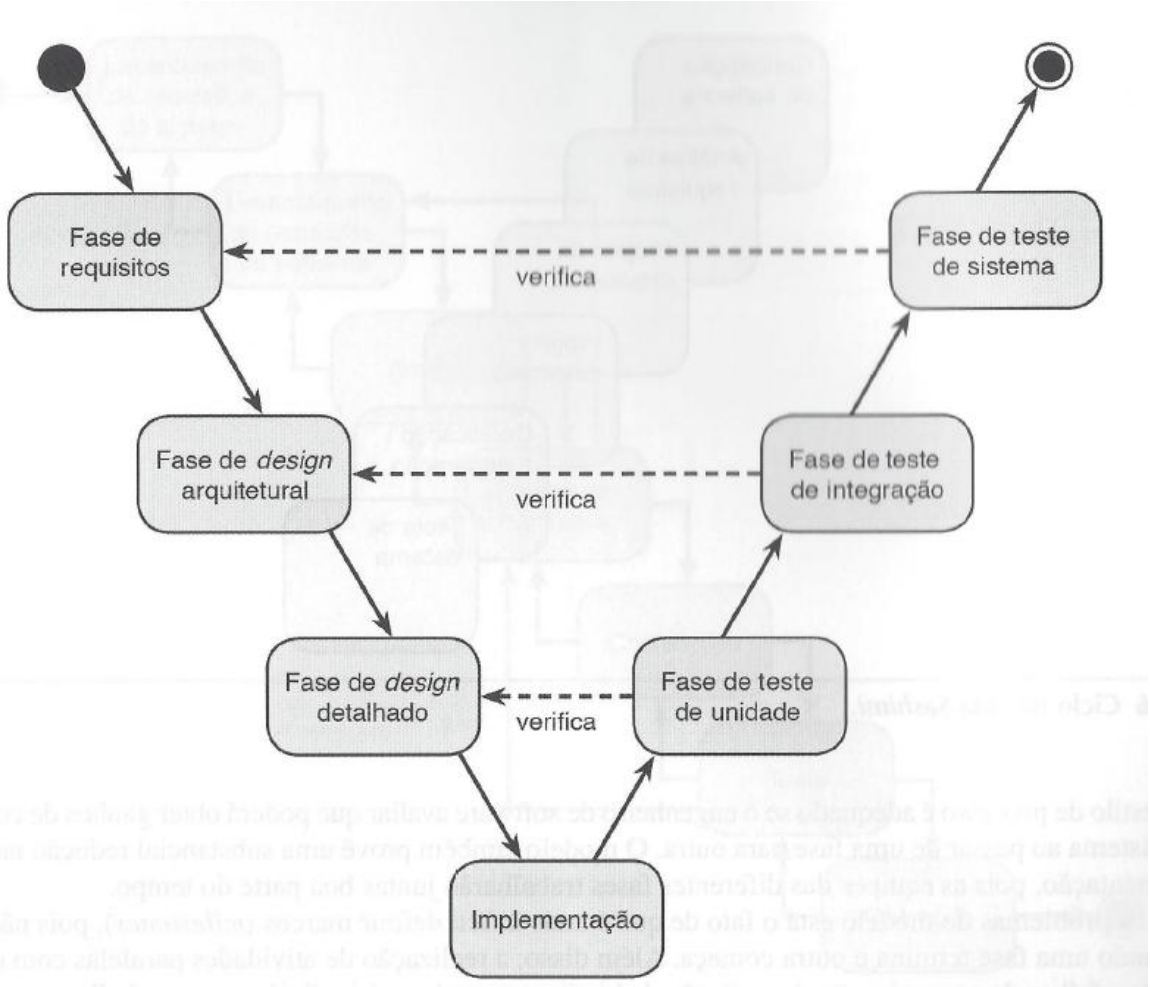
- ✓ Embora o Modelo Cascata tenha fragilidades, ele é significativamente melhor do que uma abordagem casual ao desenvolvimento de software

Variações do Modelo Cascata (Wazlawick, 2013)



CASCATA DUPLA (Royce, 1970)

Variações do Modelo Cascata (Wazlawick, 2013)



MODELO V (Spillner, 2002)

Variações do Modelo Cascata (Wazlawick, 2013)

- Outras variações:
  - Sashimi
  - Modelo W
  - Cascata com subprojetos

O Modelo Cascata

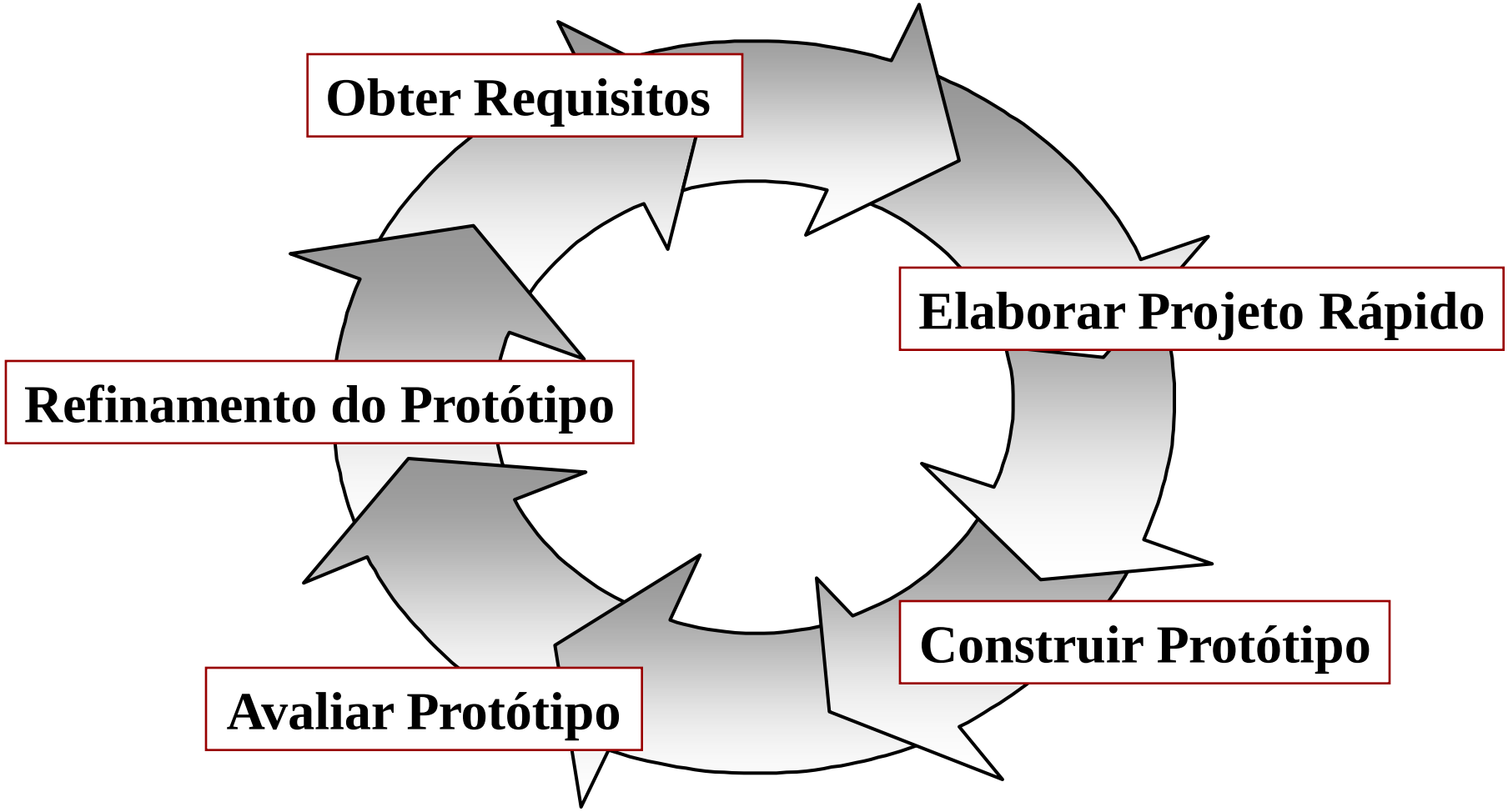
- O modelo Cascata trouxe contribuições importantes para o processo de desenvolvimento de software:
  - Imposição de disciplina, planejamento e gerenciamento
  - a implementação do produto deve ser postergada até que os objetivos tenham sido completamente entendidos

O Modelo de Prototipação

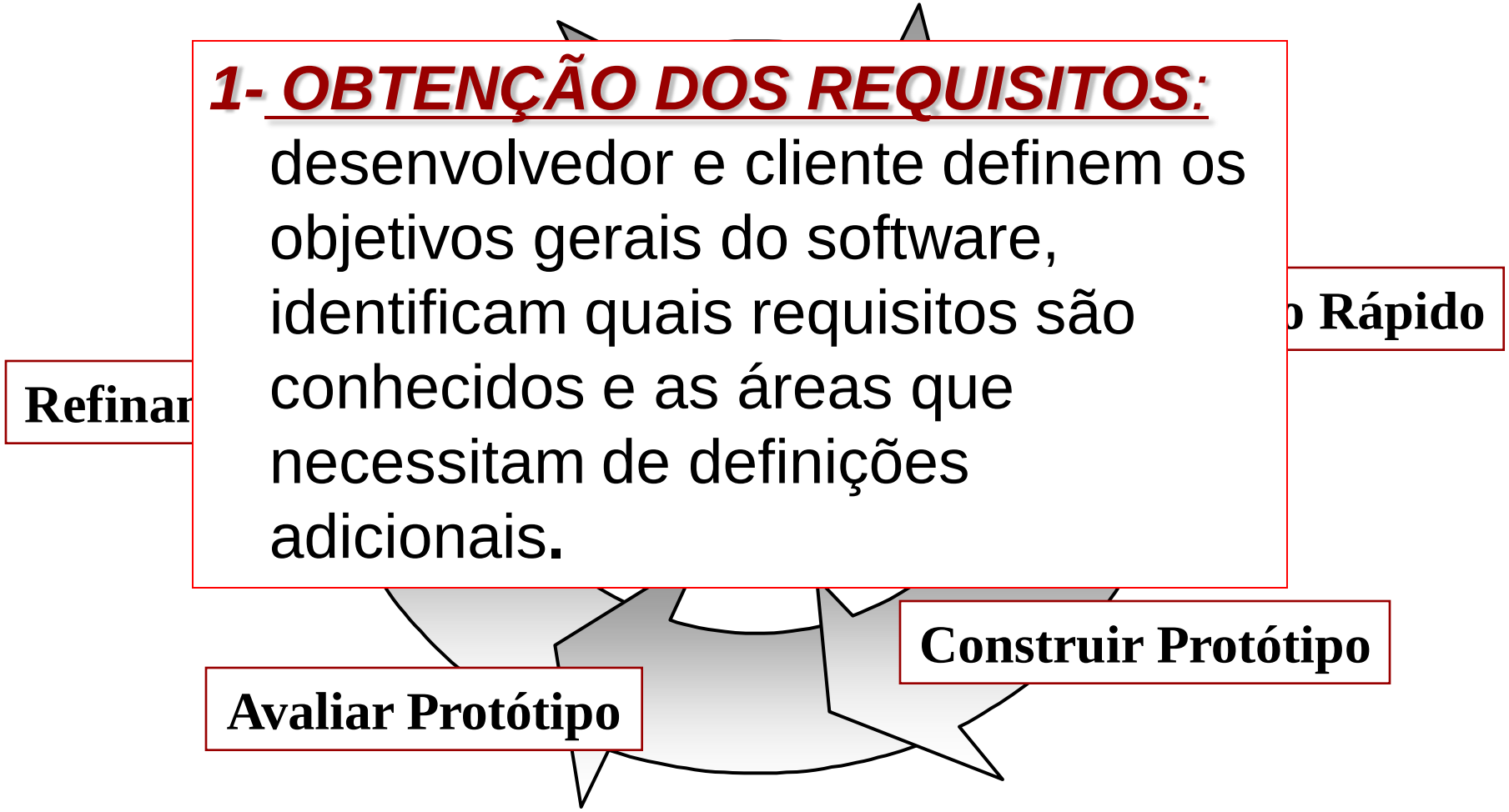
- o objetivo é entender os requisitos do usuário e, assim, obter uma melhor definição dos requisitos do sistema.
- possibilita que o desenvolvedor crie um modelo (protótipo)do software que deve ser construído
- apropriado para quando o cliente não definiu detalhadamente os requisitos.

O Paradigma de Prototipação

para obtenção dos requisitos



O Paradigma de Prototipação
para obtenção dos requisitos

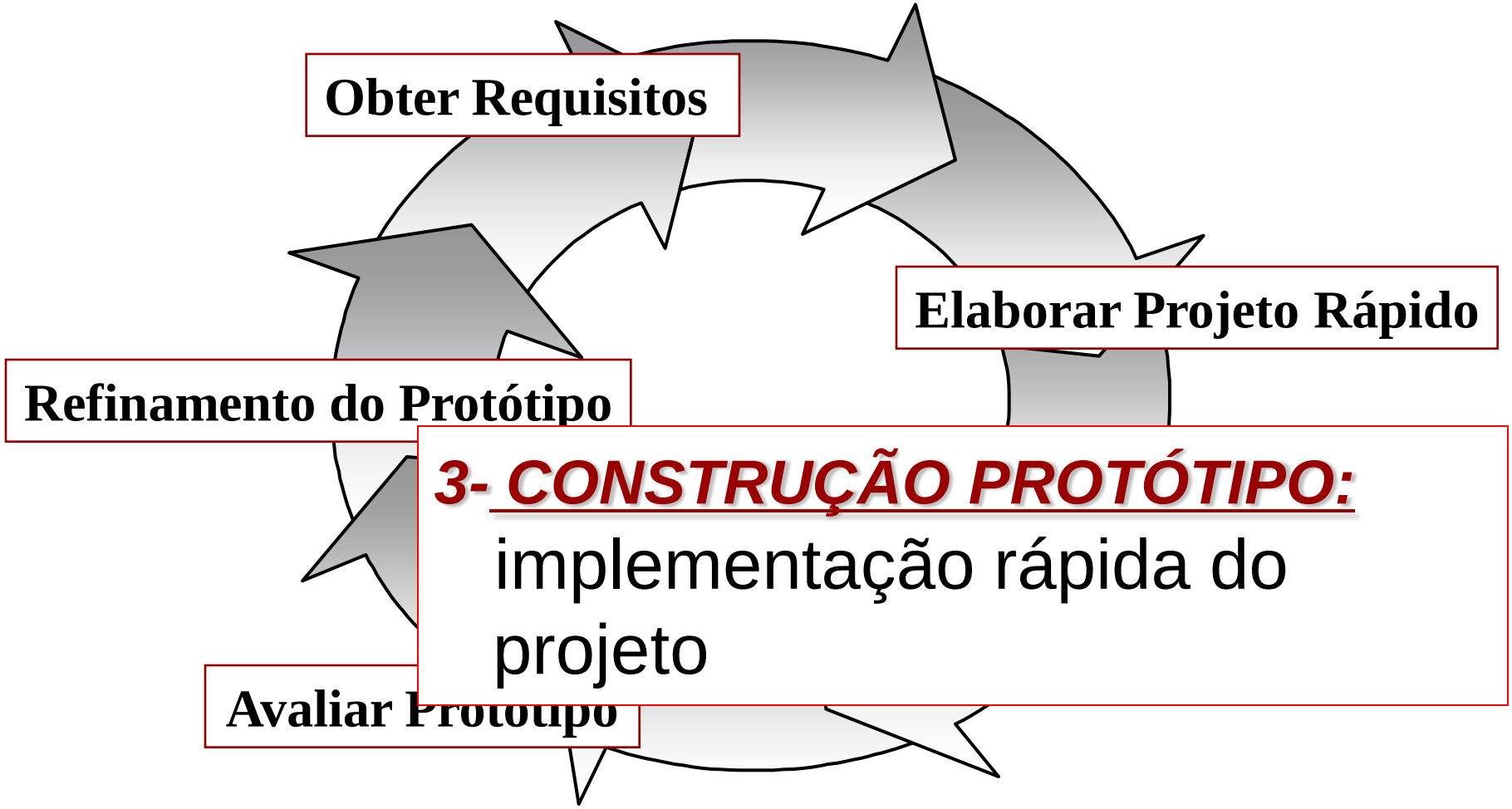


O Paradigma de Prototipação
para obtenção dos requisitos

2- PROJETO RÁPIDO:

representação dos aspectos do software que são visíveis ao usuário (abordagens de entrada e formatos de saída)

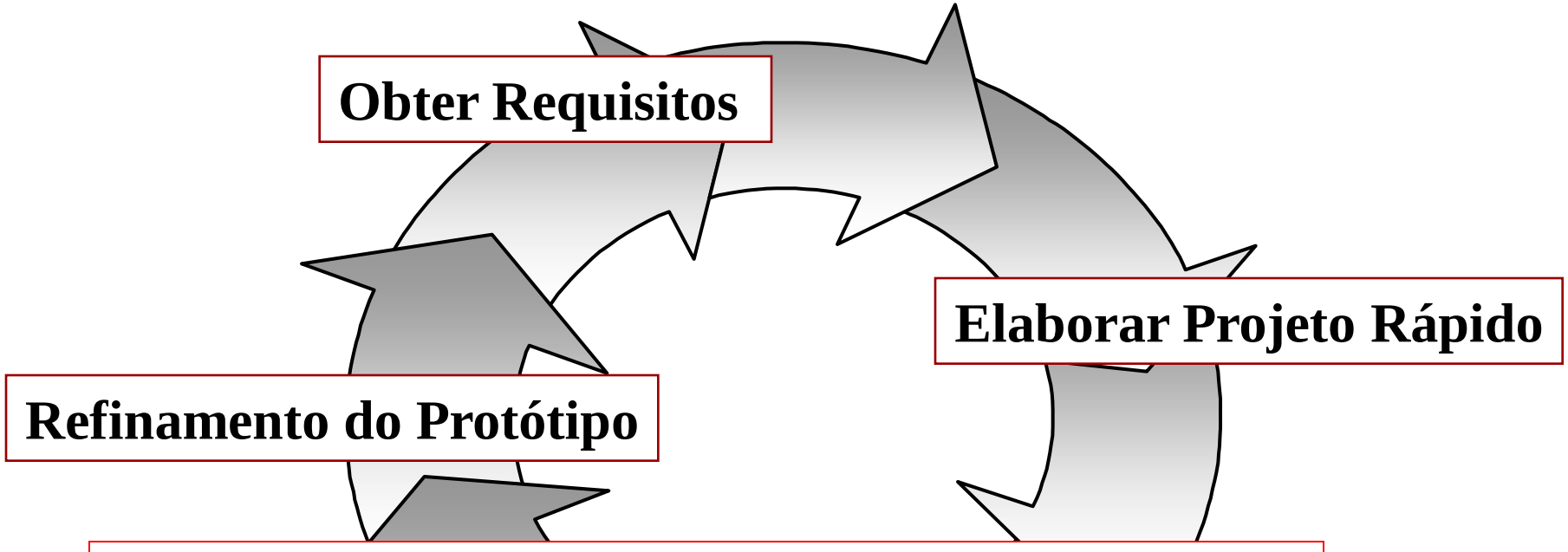
O Paradigma de Prototipação
para obtenção dos requisitos



3- CONSTRUÇÃO PROTÓTIPO:

implementação rápida do projeto

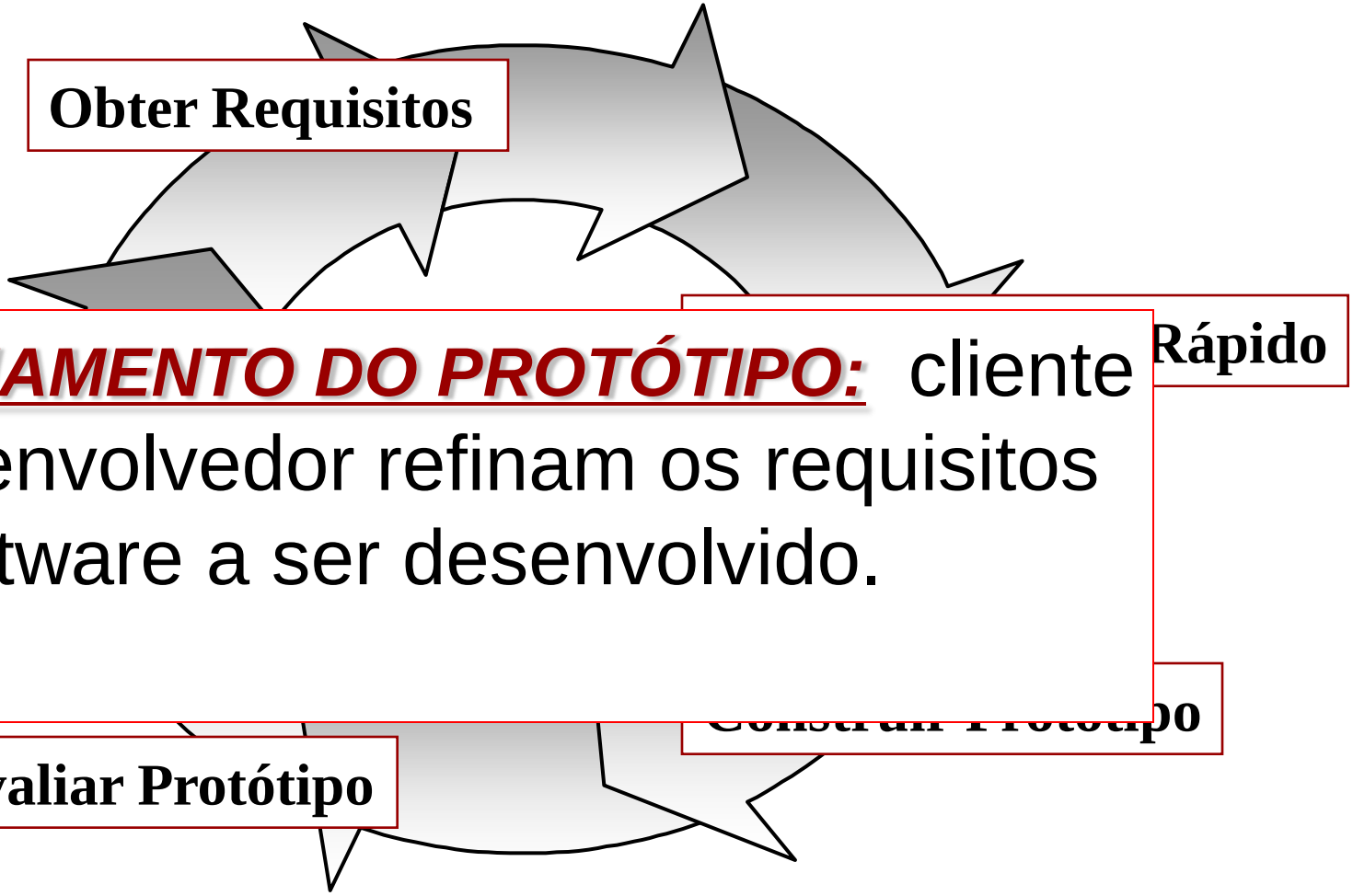
O Paradigma de Prototipação
para obtenção dos requisitos



4- AVALIAÇÃO DO PROTÓTIPO:

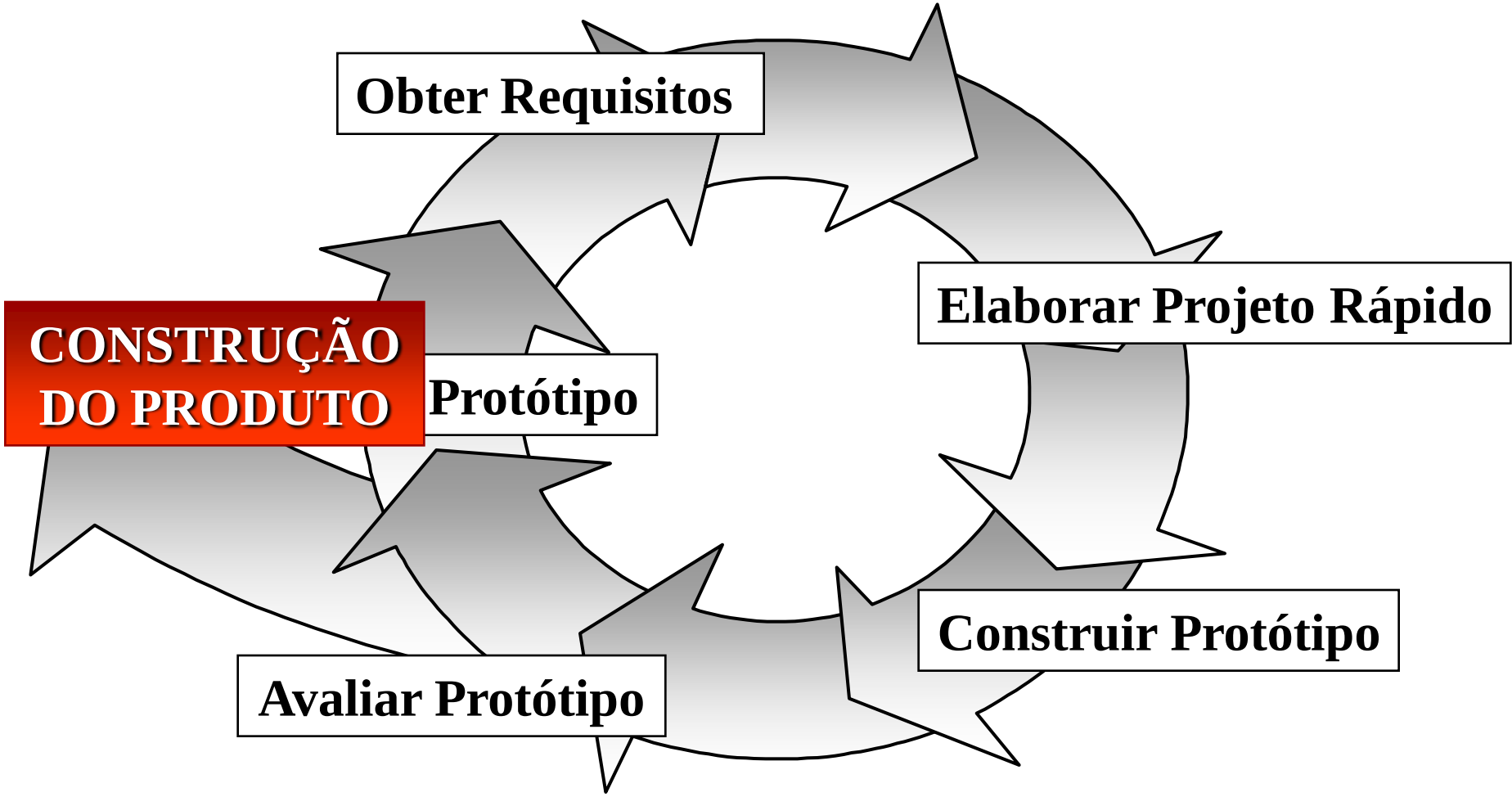
cliente e desenvolvedor avaliam o protótipo

O Paradigma de Prototipação
para obtenção dos requisitos

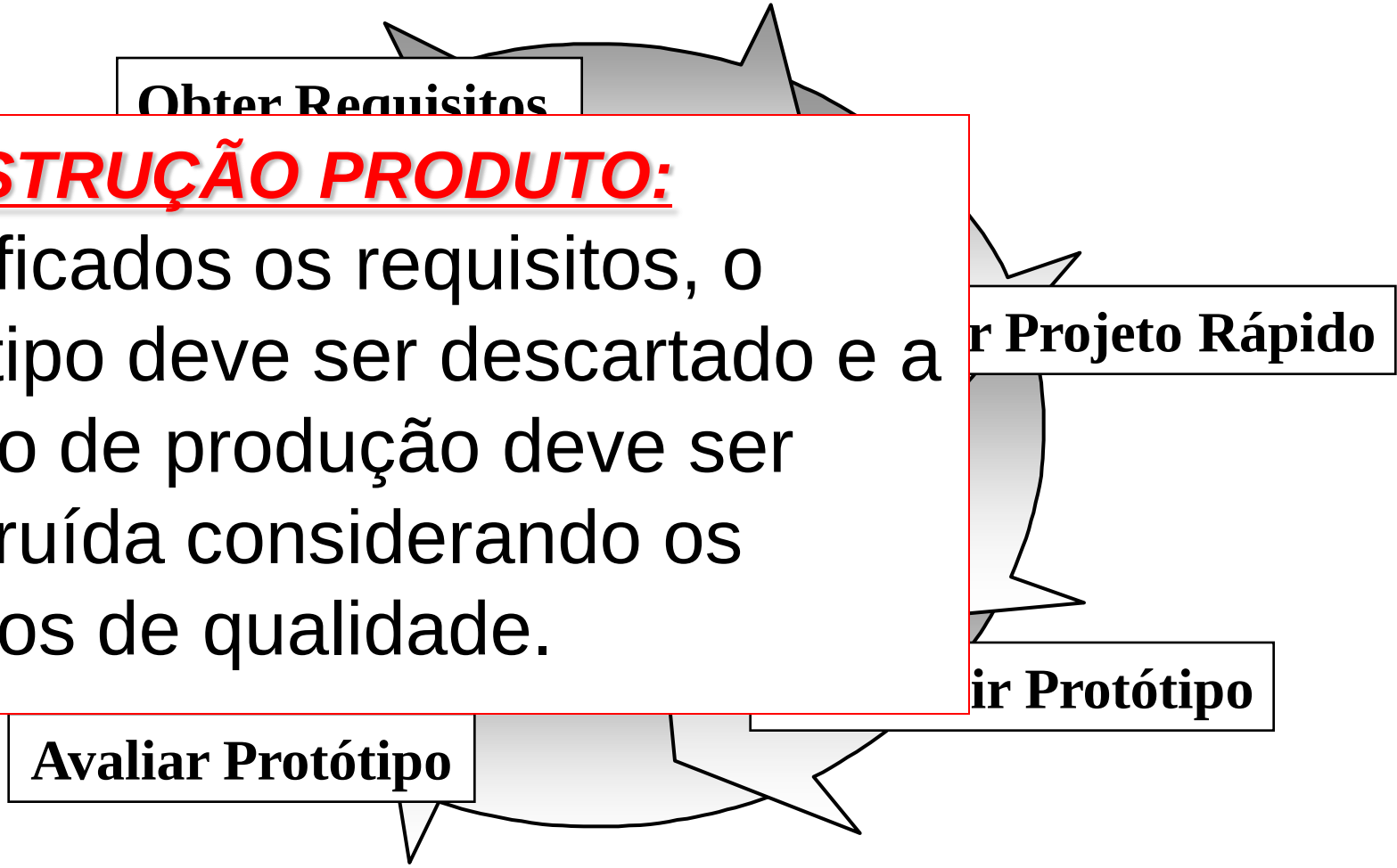


5- REFINAMENTO DO PROTÓTIPO: cliente e desenvolvedor refinam os requisitos do software a ser desenvolvido.

O Paradigma de Prototipação
para obtenção dos requisitos



O Paradigma de Prototipação
para obtenção dos requisitos



6- CONSTRUÇÃO PRODUTO:

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

- cliente não sabe que o software que ele vê não considerou, durante o desenvolvimento, a qualidade global e a manutenibilidade a longo prazo
- desenvolvedor freqüentemente faz uma implementação comprometida (utilizando o que está disponível) com o objetivo de produzir rapidamente um protótipo

Comentários sobre o Paradigma de Prototipação

- ainda que possam ocorrer problemas, a prototipação é um ciclo de vida eficiente.
- a chave é definir-se as regras do jogo logo no começo.
- o cliente e o desenvolvedor devem ambos concordar que o protótipo seja construído para servir como um mecanismo a fim de definir os requisitos

Comentários sobre o Paradigma de Prototipação

- Prototipação descartável X Prototipação evolutiva (ou evolucionária)

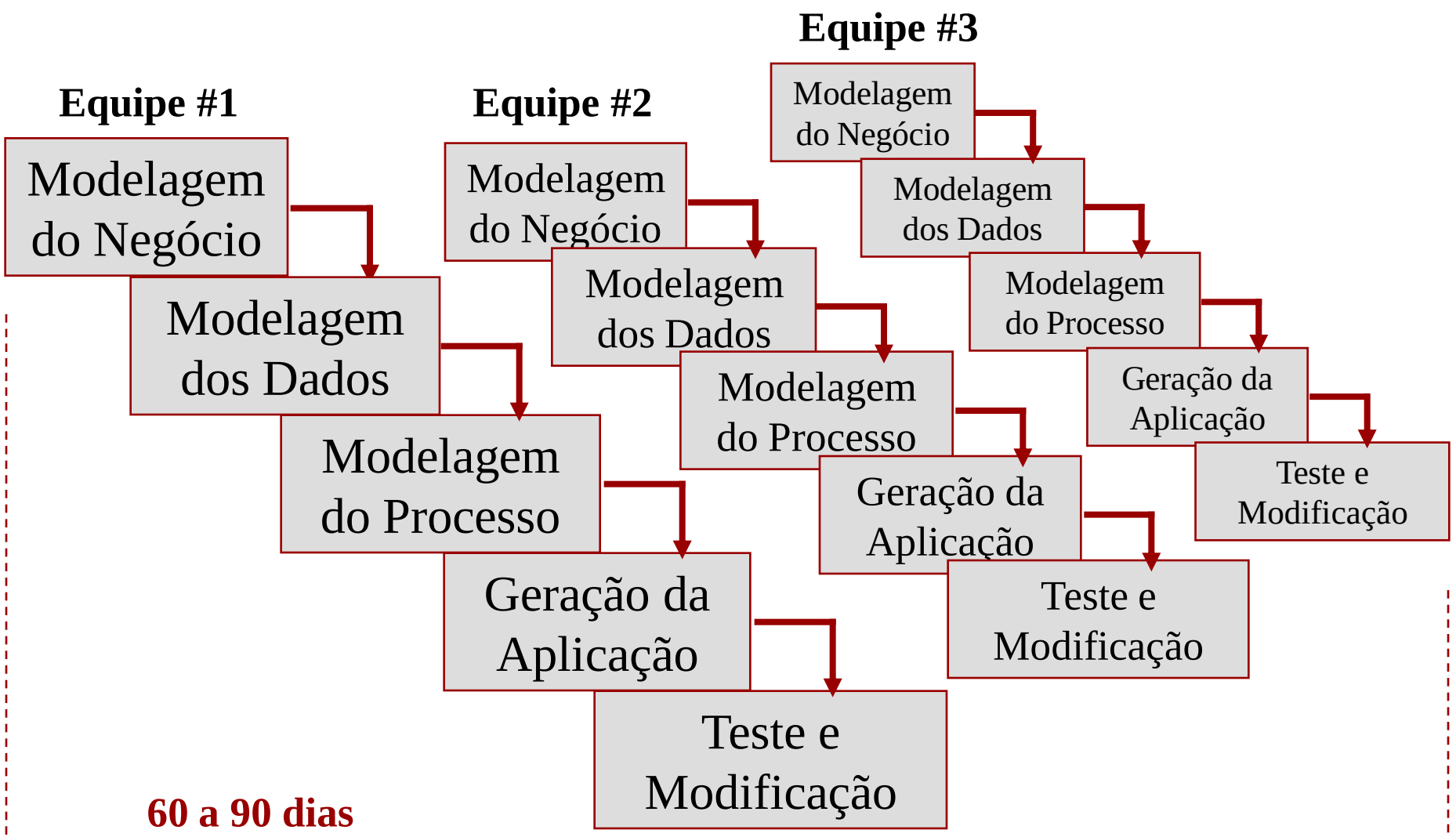
O Modelo RAD

- *RAD* (*Rapid Application Development*) é um modelo sequencial linear que enfatiza um ciclo de desenvolvimento extremamente curto
- O desenvolvimento rápido é obtido usando uma abordagem de construção baseada em componentes.

O Modelo RAD

- Os requisitos devem ser bem entendidos e o alcance do projeto restrito
- O modelo *RAD* é usado principalmente para aplicações de sistema de informação
- Cada função principal pode ser direcionada para uma equipe RAD separada e então integrada para formar o todo.

O Modelo RAD


Equipe #1
Modelagem do Negócio
Modelagem dos Dados
Modelagem do Processo
Geração da Aplicação
Teste e Modificação
Equipe #2
Modelagem do Negócio
Modelagem dos Dados
Modelagem do Processo
Geração da Aplicação
Teste e Modificação
Equipe #3
Modelagem do Negócio
Modelagem dos Dados
Modelagem do Processo
Geração da Aplicação
Teste e Modificação
60 a 90 dias

O Modelo RAD

Desvantagens:

- Exige recursos humanos suficientes para todas as equipes
- Exige que desenvolvedores e clientes estejam comprometidos com as atividades de “fogo-rápido” a fim de terminar o projeto num prazo curto

O Modelo RAD

- Nem todos os tipos de aplicação são apropriadas para o RAD:
  - Deve ser possível a modularização efetiva da aplicação
  - se alto desempenho é uma característica e o desempenho é obtido sintonizando as interfaces dos componentes do sistema, a abordagem RAD pode não funcionar

Modelos de Processo de Software

- *O Modelo Sequencial Linear*
  - *também chamado Modelo Cascata*
- *O Modelo de Prototipação*
- *O Modelo RAD (Rapid Application Development)*
- Modelos Evolutivos de Processo de Software
  - O *Modelo Incremental*
    - *O Modelo Ágil*
    - *O Processo Unificado*
  - O *Modelo Espiral*
  - O *Modelo de Montagem de Componentes*
  - O *Modelo de Desenvolvimento Concorrente*
- *Modelo de Métodos Formais*
- *Técnicas de Quarta Geração*

Modelos Evolutivos de Processo

- Existem situações em que a engenharia de software necessita de um modelo de processo que possa acomodar um produto que evolui com o tempo.

Modelos Evolutivos de Processo

- quando os requisitos de produto e de negócio mudam conforme o desenvolvimento procede
- quando há 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 de Processo

- modelos evolutivos são iterativos
- possibilitam o desenvolvimento de versões cada vez mais completas do software

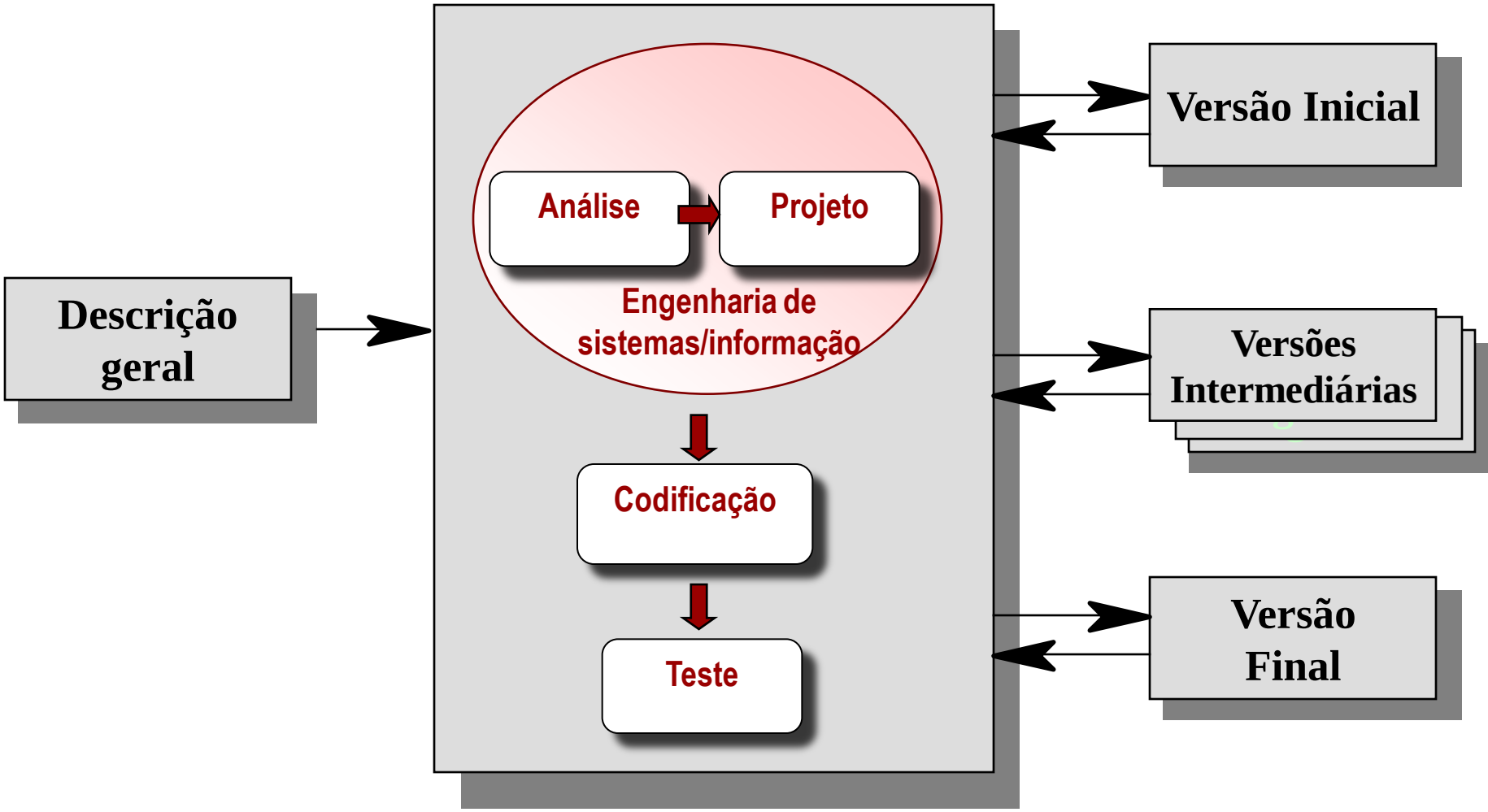
Modelos de Processo de Software

- *O Modelo Sequencial Linear*
  - *também chamado Modelo Cascata*
- *O Modelo de Prototipação*
- *O Modelo RAD (Rapid Application Development)*
- Modelos Evolutivos de Processo de Software
  - O *Modelo Incremental*
    - *O Modelo Ágil*
    - *O Processo Unificado*
  - O *Modelo Espiral*
  - O *Modelo de Montagem de Componentes*
  - O *Modelo de Desenvolvimento Concorrente*
- *Modelo de Métodos Formais*
- *Técnicas de Quarta Geração*

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

- o modelo incremental é mais apropriado para sistemas pequenos
- As novas versões podem ser planejadas de modo que os riscos técnicos possam ser administrados (Ex. disponibilidade de determinado hardware)

Modelos de Processo de Software

- *O Modelo Sequencial Linear*
  - *também chamado Modelo Cascata*
- *O Modelo de Prototipação*
- *O Modelo RAD (Rapid Application Development)*
- Modelos Evolutivos de Processo de Software
  - O *Modelo Incremental*
    - *O Modelo Ágil*
    - *O Processo Unificado*
  - O *Modelo Espiral*
  - O *Modelo de Montagem de Componentes*
  - O *Modelo de Desenvolvimento Concorrente*
- *Modelo de Métodos Formais*
- *Técnicas de Quarta Geração*

O Modelo Espiral

- Engloba as melhores características do Modelo Cascata e da Prototipação, adicionando um novo elemento: a Análise de Riscos.
  - Risco: algo que pode dar errado.
  - Segue a abordagem de passos sistemáticos do Modelo Cascata, incorporando-os numa estrutura iterativa.
  - É dividido em uma série de regiões, tipicamente de 3 a 6, que delimitam atividades de arcabouço (*framework activities*)
  - Usa a Prototipação em qualquer etapa da evolução do produto, como mecanismo de redução de riscos.

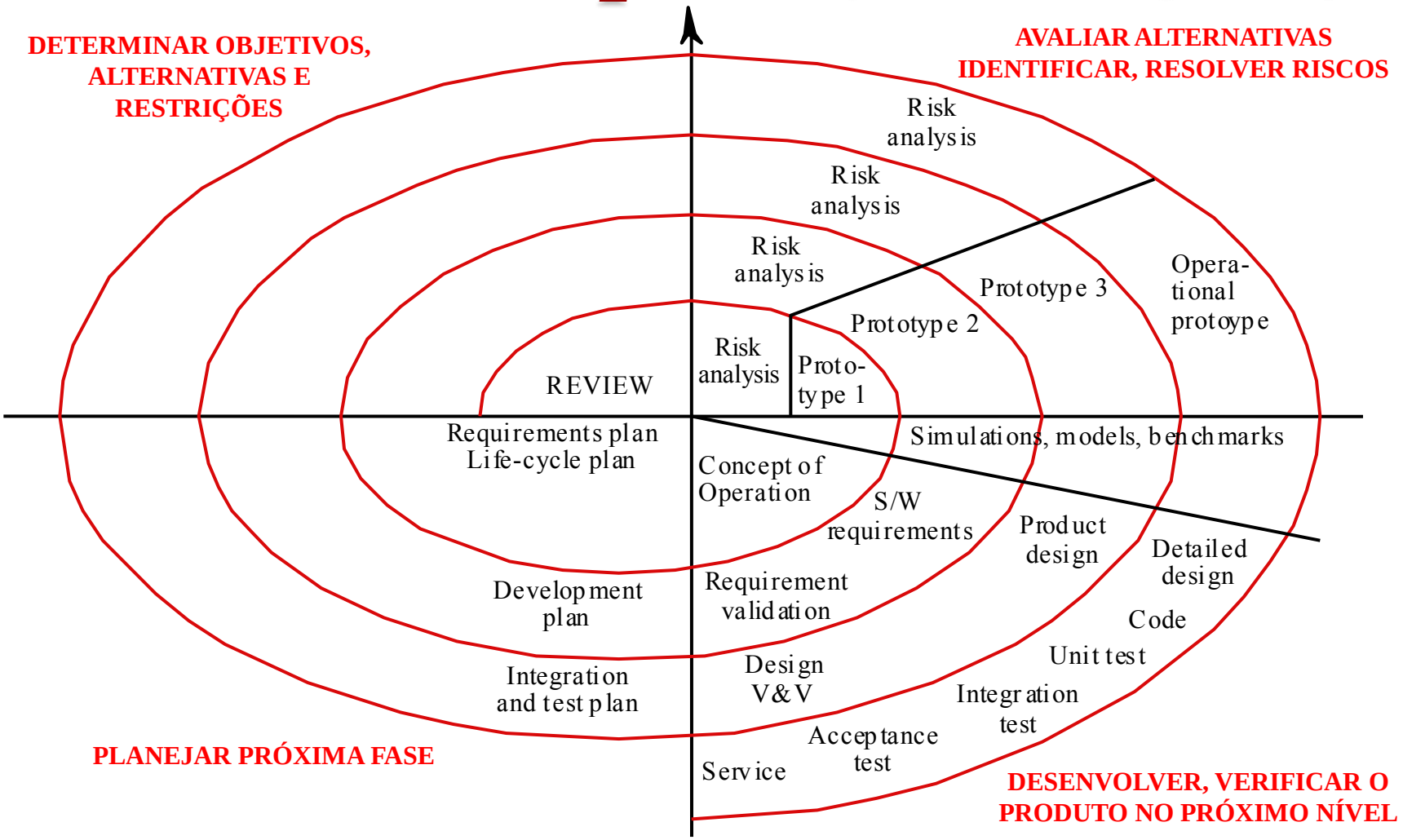
Modelo Espiral

- É uma abordagem realista para o desenvolvimento de sistemas e software de grande porte.
- Exige a consideração direta dos riscos técnicos em todos os estágios do projeto.
  - Se aplicado adequadamente, pode reduzir os riscos antes que eles fiquem problemáticos.

O Modelo Espiral

- Funcionamento:
  - Para cada volta na espiral:
    - Determinar os objetivos, alternativas e restrições relacionadas à iteração que vai se iniciar
    - Identificar e resolver os riscos relacionados
    - Avaliar alternativas disponíveis. Podem ser feitos protótipos para analisar a viabilidade de diferentes alternativas
    - Desenvolver os artefatos relacionados à iteração corrente e valida-los
    - Planejar a próxima iteração
    - Obter concordância em relação ao planejamento

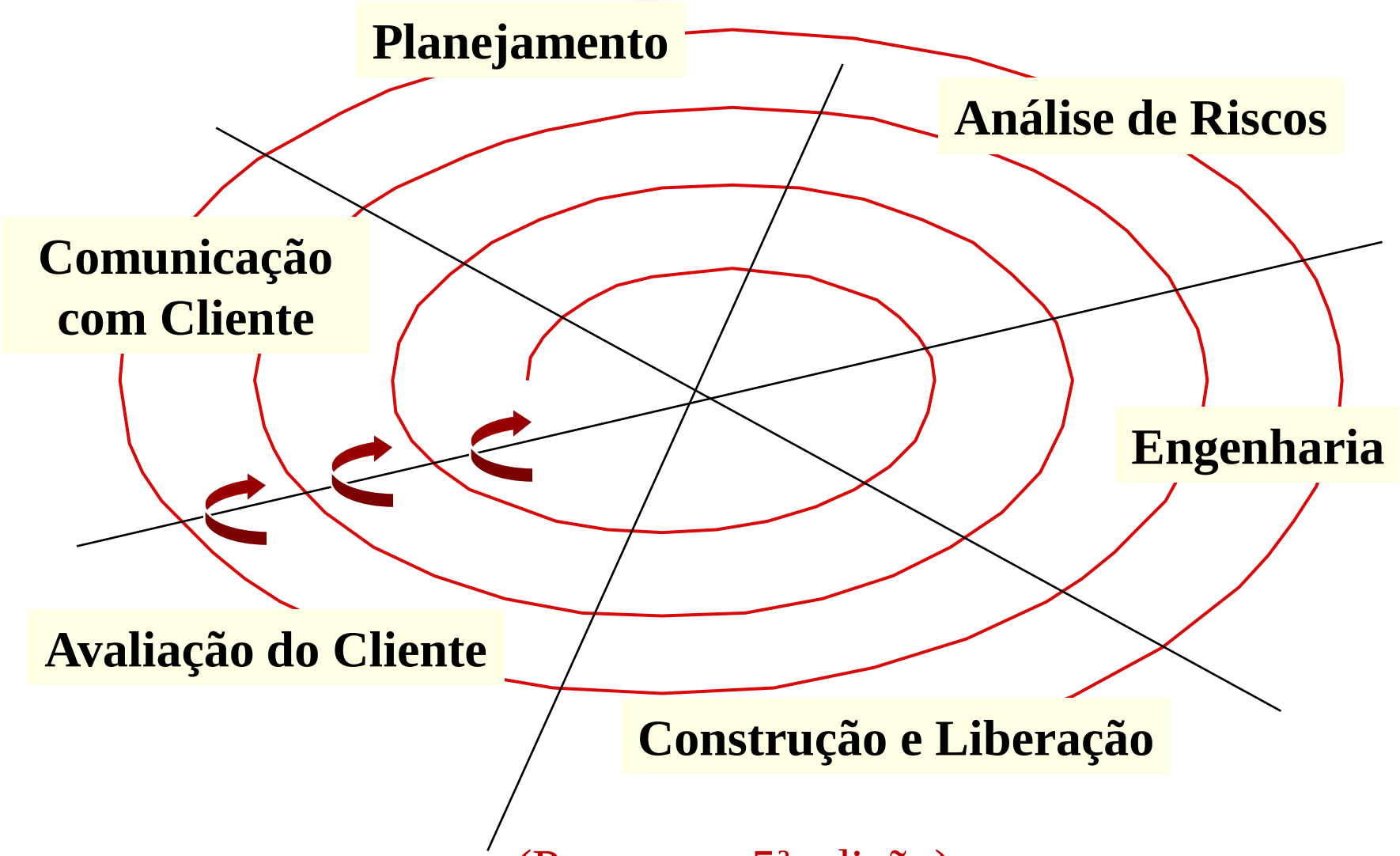
O Modelo Espiral (com 4 regiões)



(Sommerville, 9ª edição)

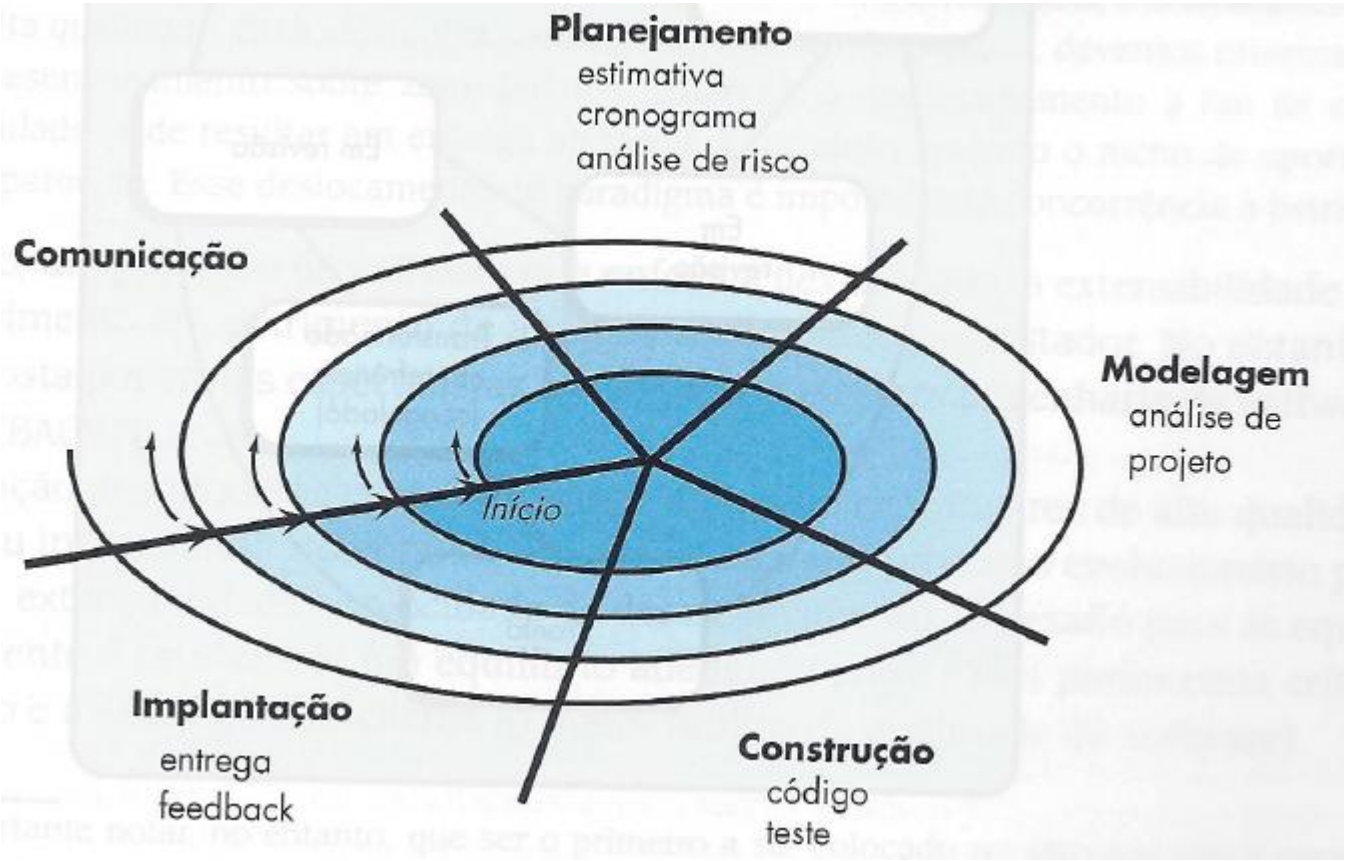
(Wazlawick, 2013)

O Modelo Espiral (com 6 regiões)



(Pressman, 5ª edição)

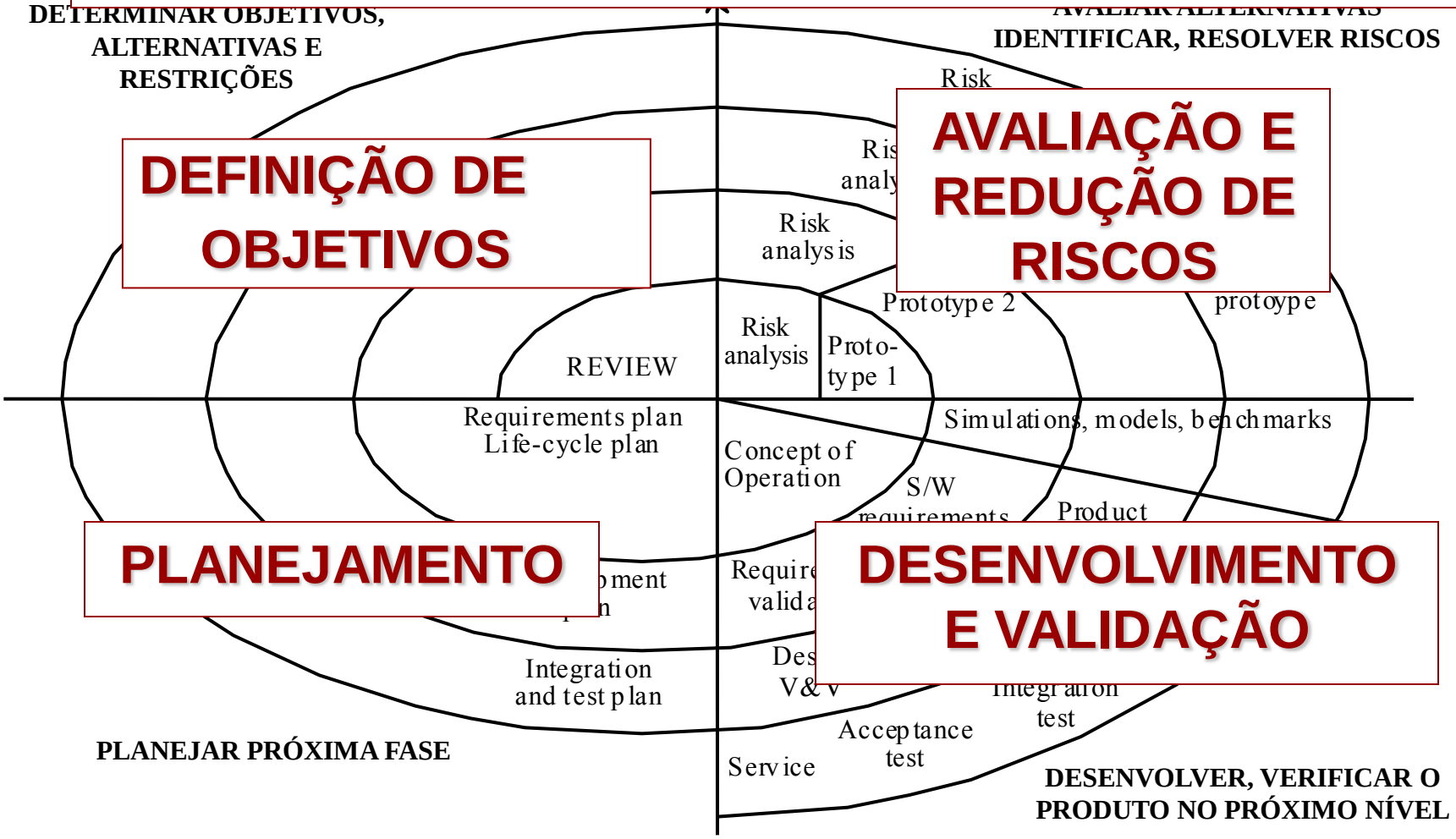
O Modelo Espiral (com 5 regiões)



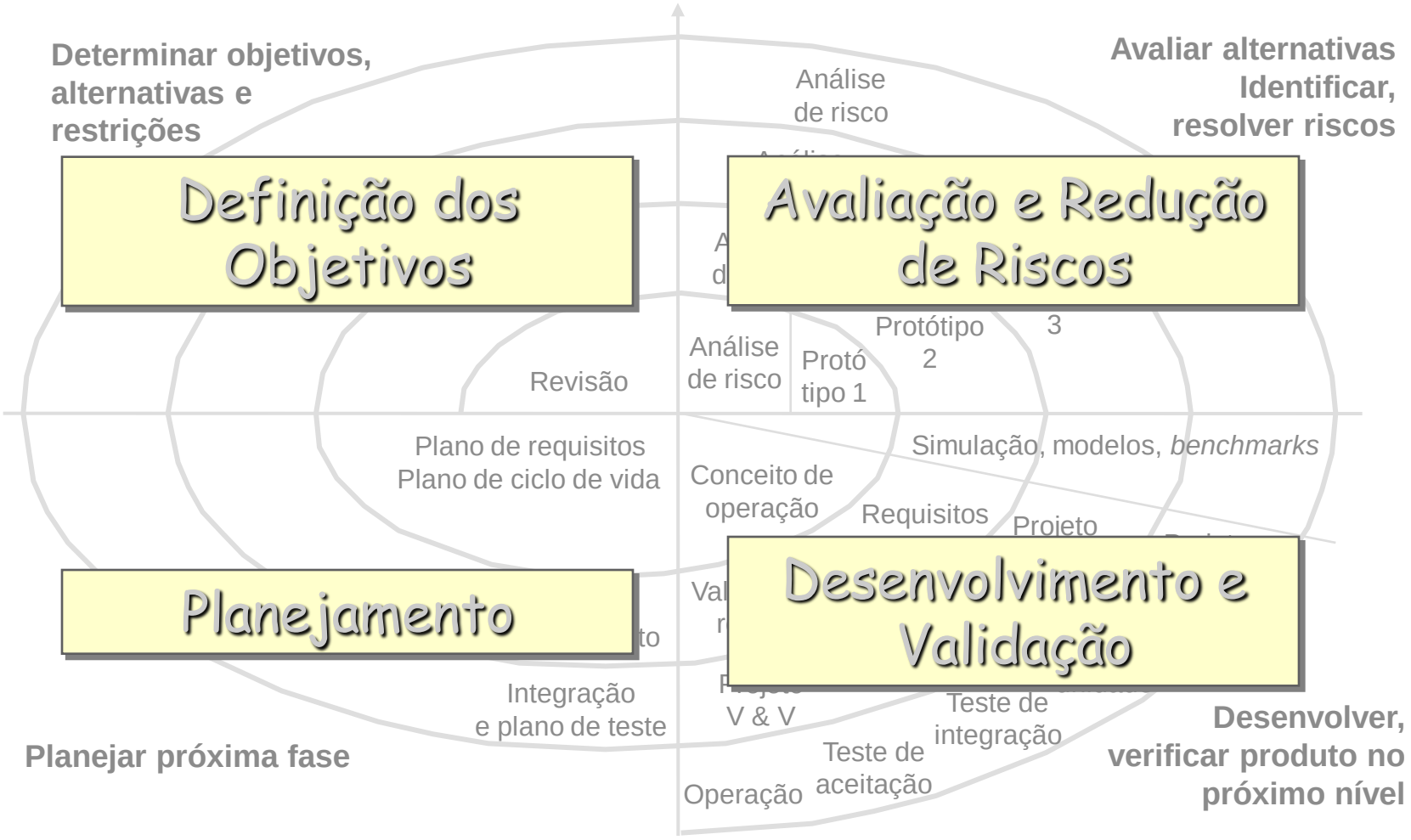
(Pressman, 6ª edição)

O Modelo Espiral

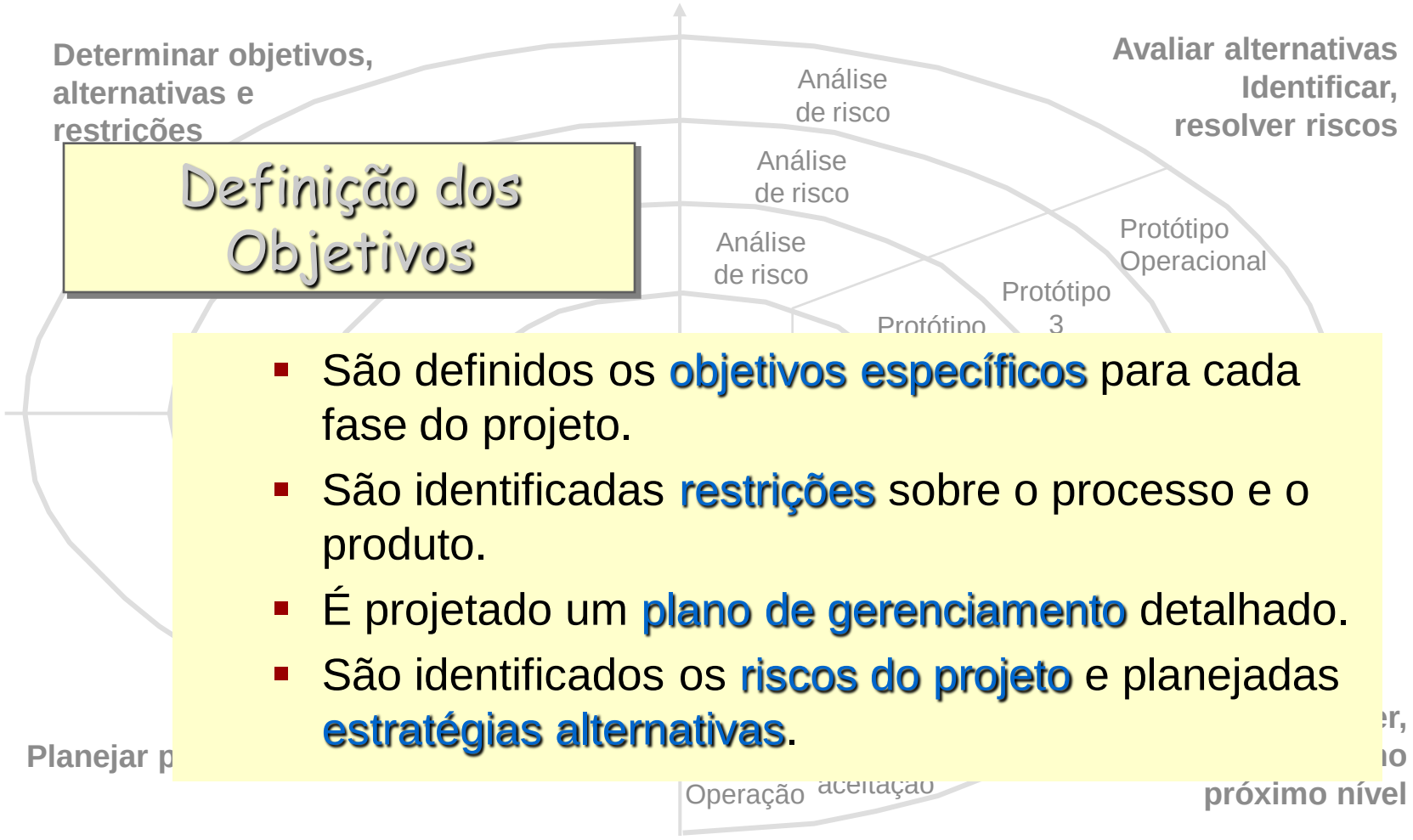
- No modelo com 4 regiões, cada "loop" do espiral é dividido em 4 partes



O Modelo Espiral



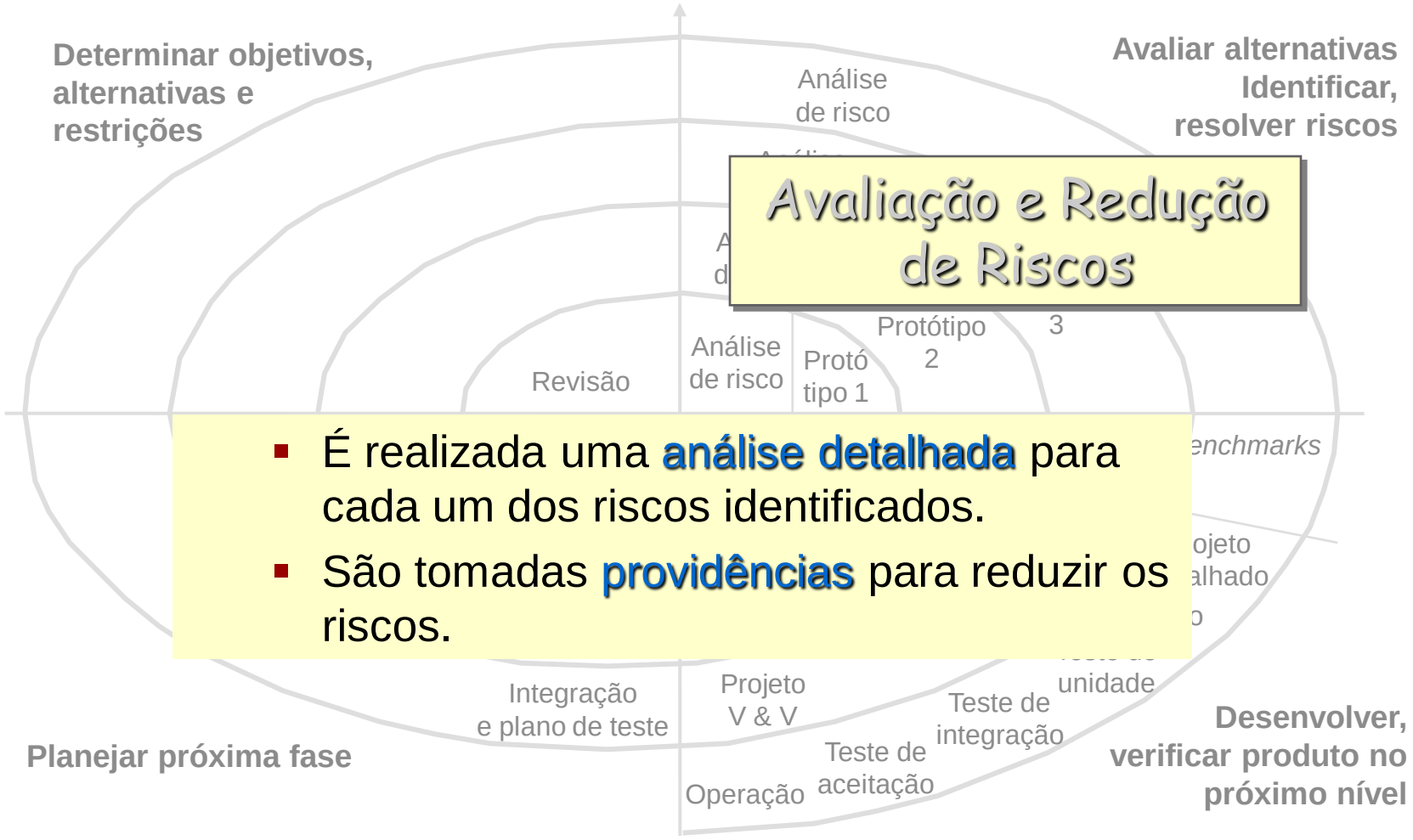
O Modelo Espiral



Definição dos Objetivos

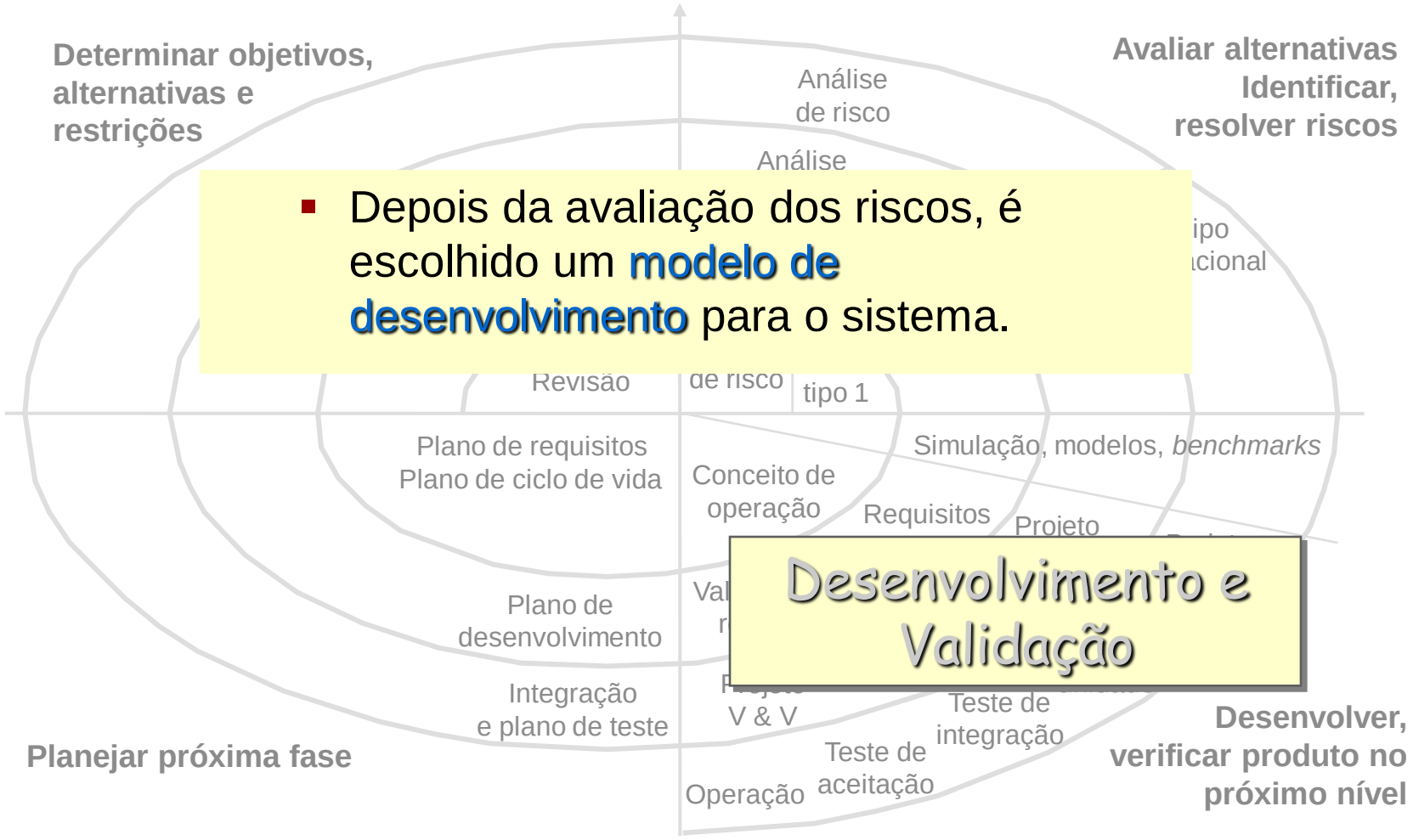
- São definidos os **objetivos específicos** para cada fase do projeto.
- São identificadas **restrições** sobre o processo e o produto.
- É projetado um **plano de gerenciamento** detalhado.
- São identificados os **riscos do projeto** e planejadas **estratégias alternativas**.

O Modelo Espiral



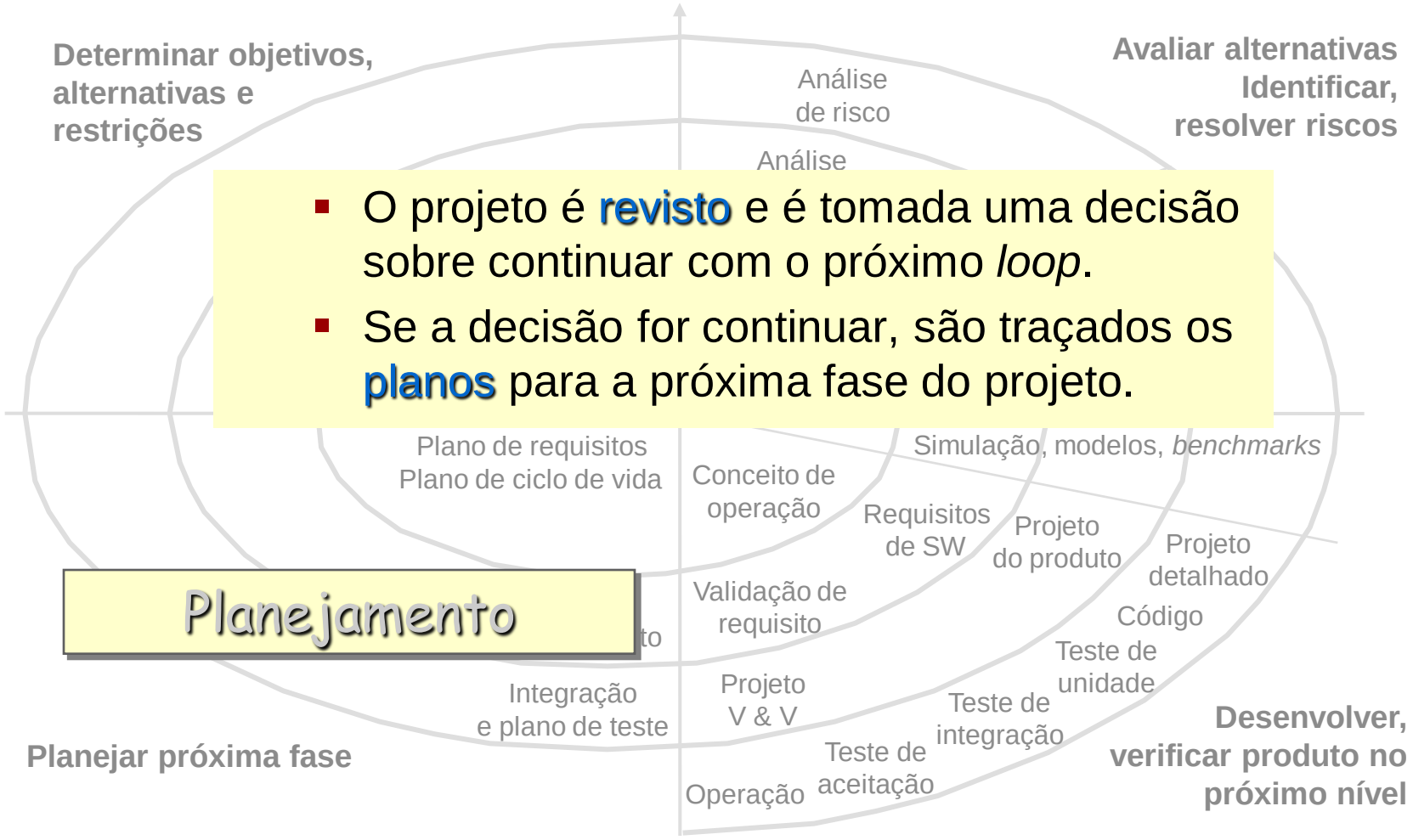
- É realizada uma **análise detalhada** para cada um dos riscos identificados.
- São tomadas **providências** para reduzir os riscos.

O Modelo Espiral



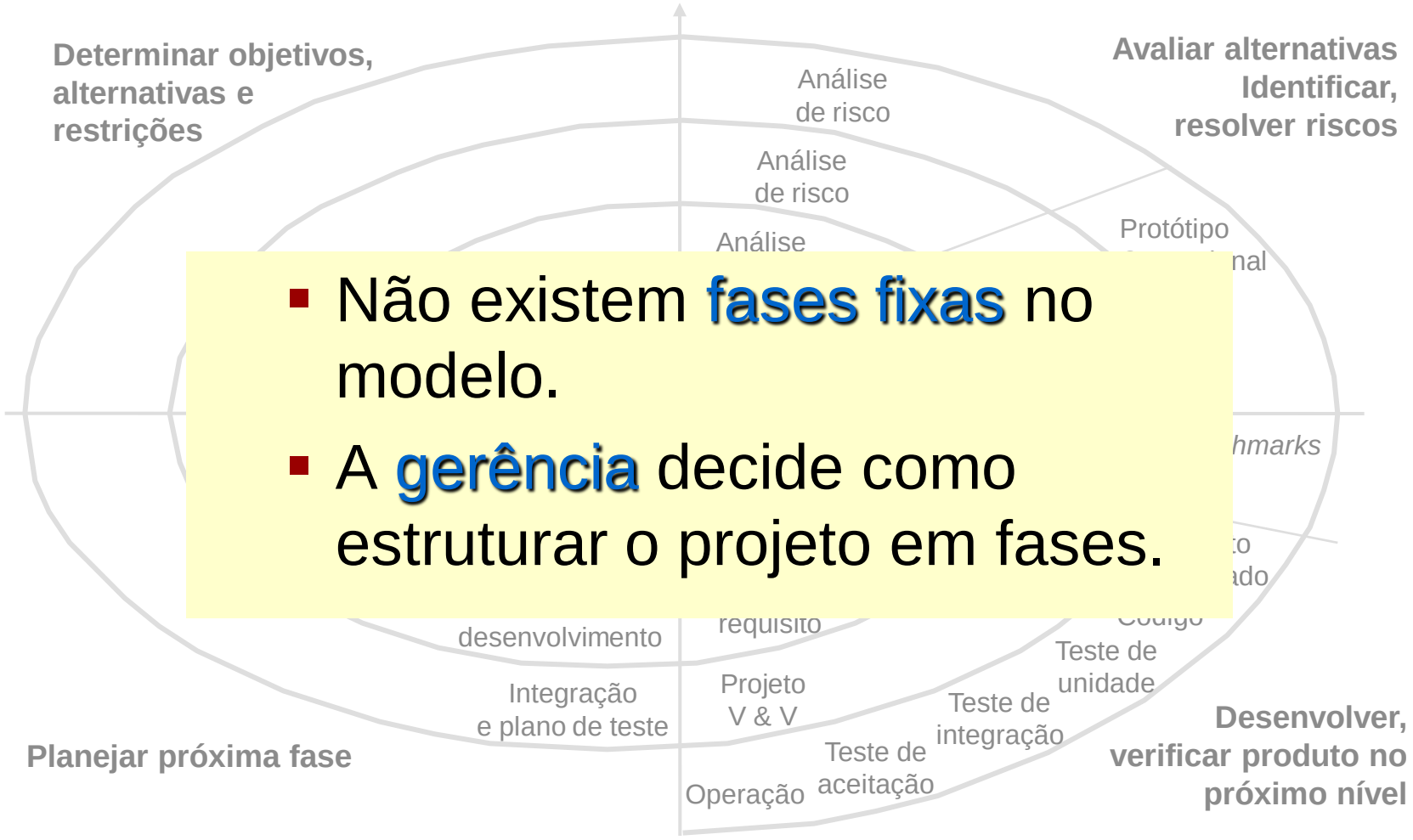
- Depois da avaliação dos riscos, é escolhido um **modelo de desenvolvimento** para o sistema.

O Modelo Espiral

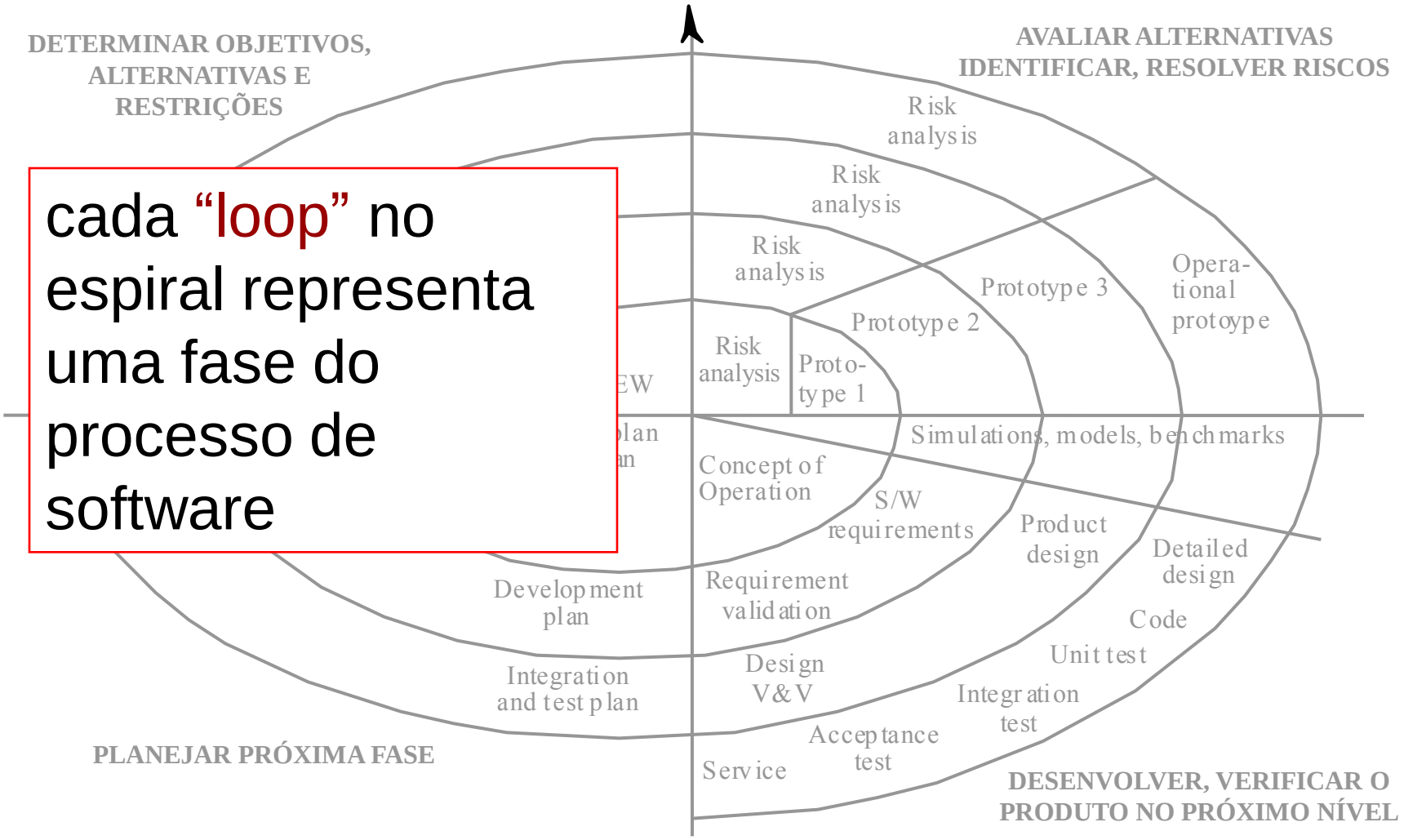


- O projeto é revisto e é tomada uma decisão sobre continuar com o próximo *loop*.
- Se a decisão for continuar, são traçados os planos para a próxima fase do projeto.

O Modelo Espiral



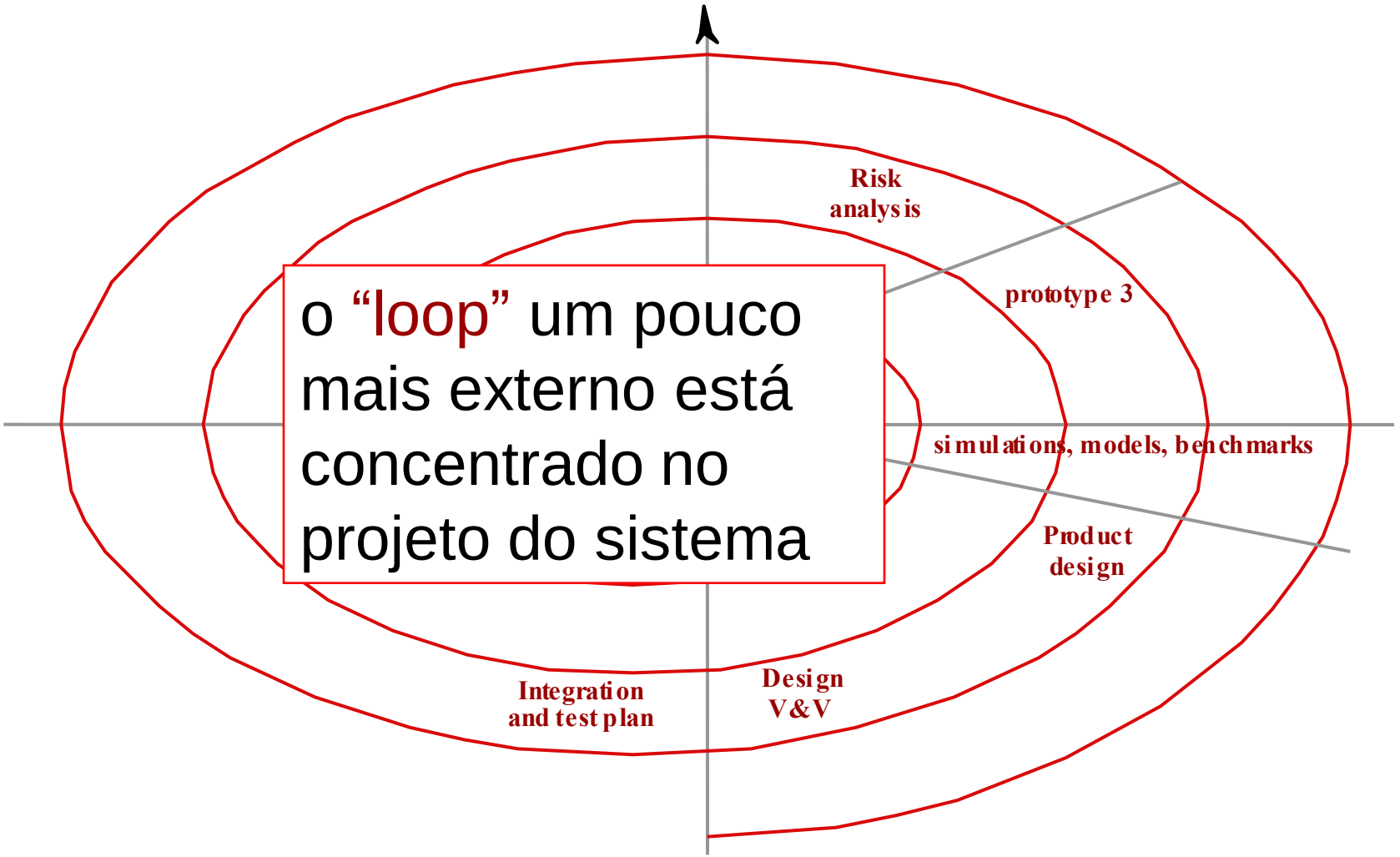
- Não existem **fases fixas** no modelo.
- A **gerência** decide como estruturar o projeto em fases.



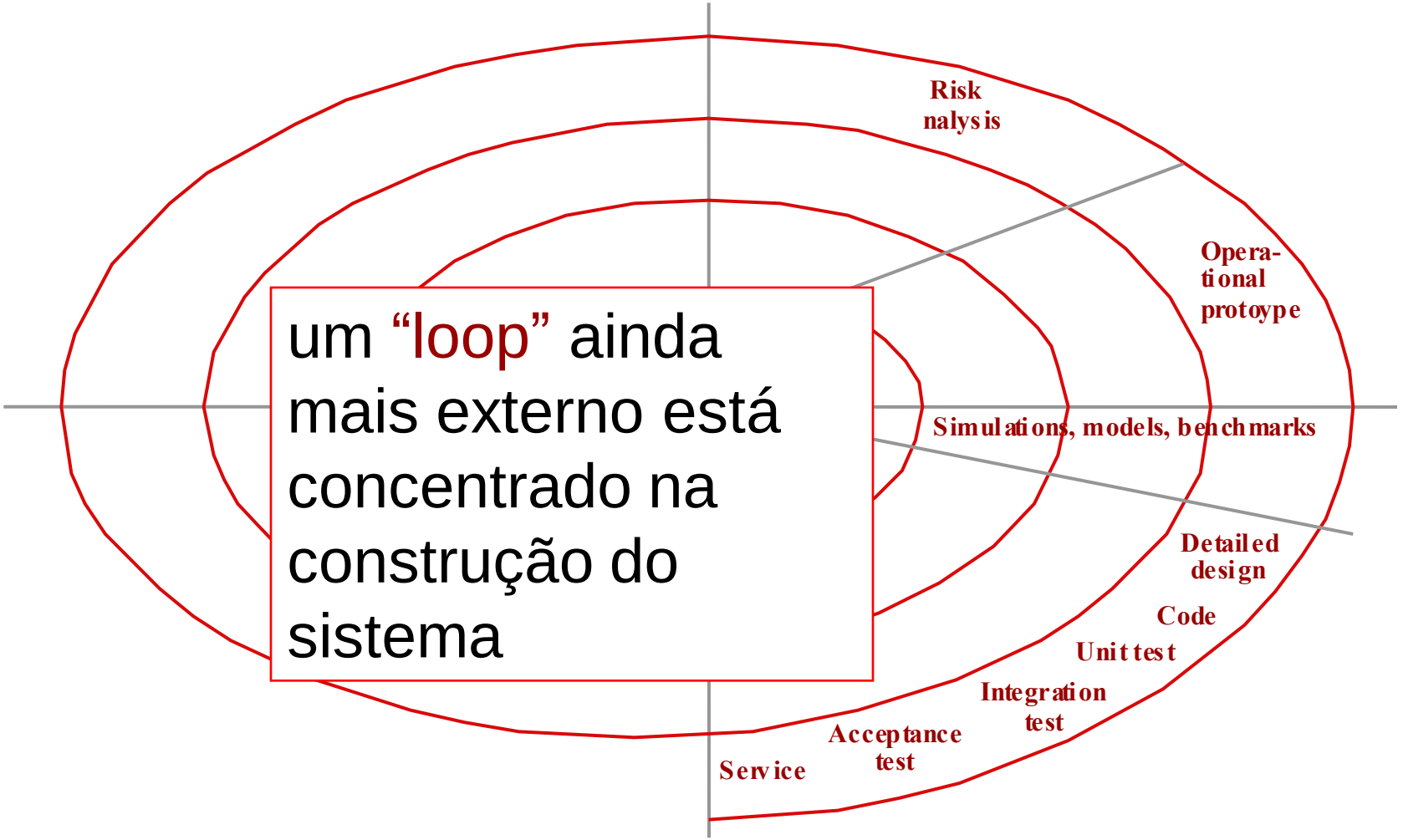
cada “loop” no espiral representa uma fase do processo de software

o “loop” mais interno está concentrado nas possibilidades do sistema

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



o “loop” um pouco mais externo está concentrado no projeto do sistema



um “loop” ainda mais externo está concentrado na construção do sistema

Comentários sobre o Ciclo de Vida em Espiral

- Se o projeto não puder ser concluído por razões técnicas, isso é descoberto cedo, antes de um grande investimento ser feito
  - As primeiras voltas da espiral são mais baratas
- Porém, exige gerência complexa e eficiente

Comentários sobre o Ciclo de Vida em Espiral

- 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
- o modelo é relativamente novo e não tem sido amplamente usado. Demorará muitos anos até que a eficácia desse modelo possa ser determinada com certeza absoluta

Atividade

Considerando os modelos de processo de software vistos até aqui, você consegue pensar em um exemplo típico de sistema adequado para cada um deles?

Justifique a escolha!!