Modelos de Processo de Software

Engenharia de Software

Profa. Dra. Elisa Yumi Nakagawa

1º semestre de 2015

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]

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 Espiral*
  - O *Modelo de Montagem de Componentes*
  - O *Modelo de Desenvolvimento Concorrente*
- *Modelo de Métodos Formais*
- *Técnicas de Quarta Geração*

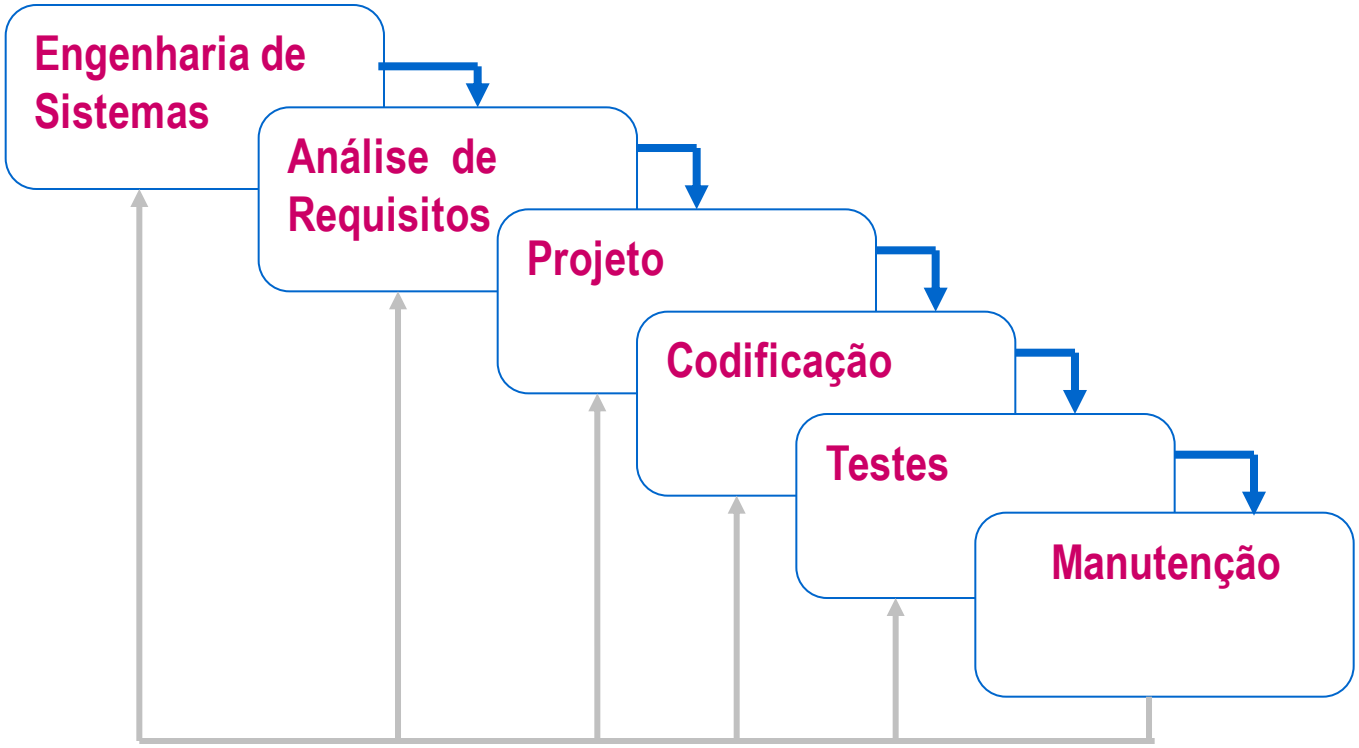
Modelos de Processo de Software

- *O Modelo Sequencial Linear*
  - *também chamado **Modelo Cascata***
- *O Paradigma de Prototipação*
- *O Modelo RAD (Rapid Application Development)*
- Modelos Evolutivos de Processo de Software
  - *O Modelo Incremental*
  - ***O Modelo Espiral***
  - *O Modelo de Montagem de Componentes*
  - *O Modelo de Desenvolvimento Concorrente*
- *Modelos 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

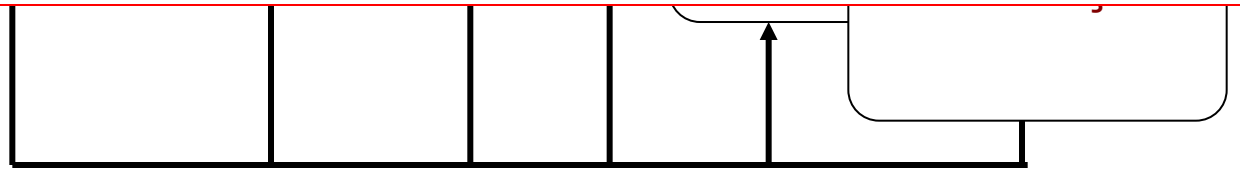
Análise de Requisitos de Software

- 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

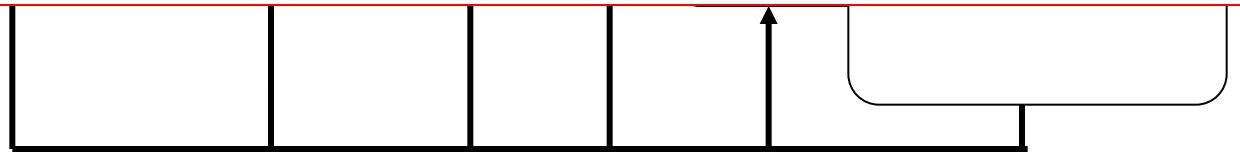
- tradução dos requisitos do software para um conjunto de representações que podem ser avaliadas quanto à qualidade, antes que a codificação se inicie



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

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

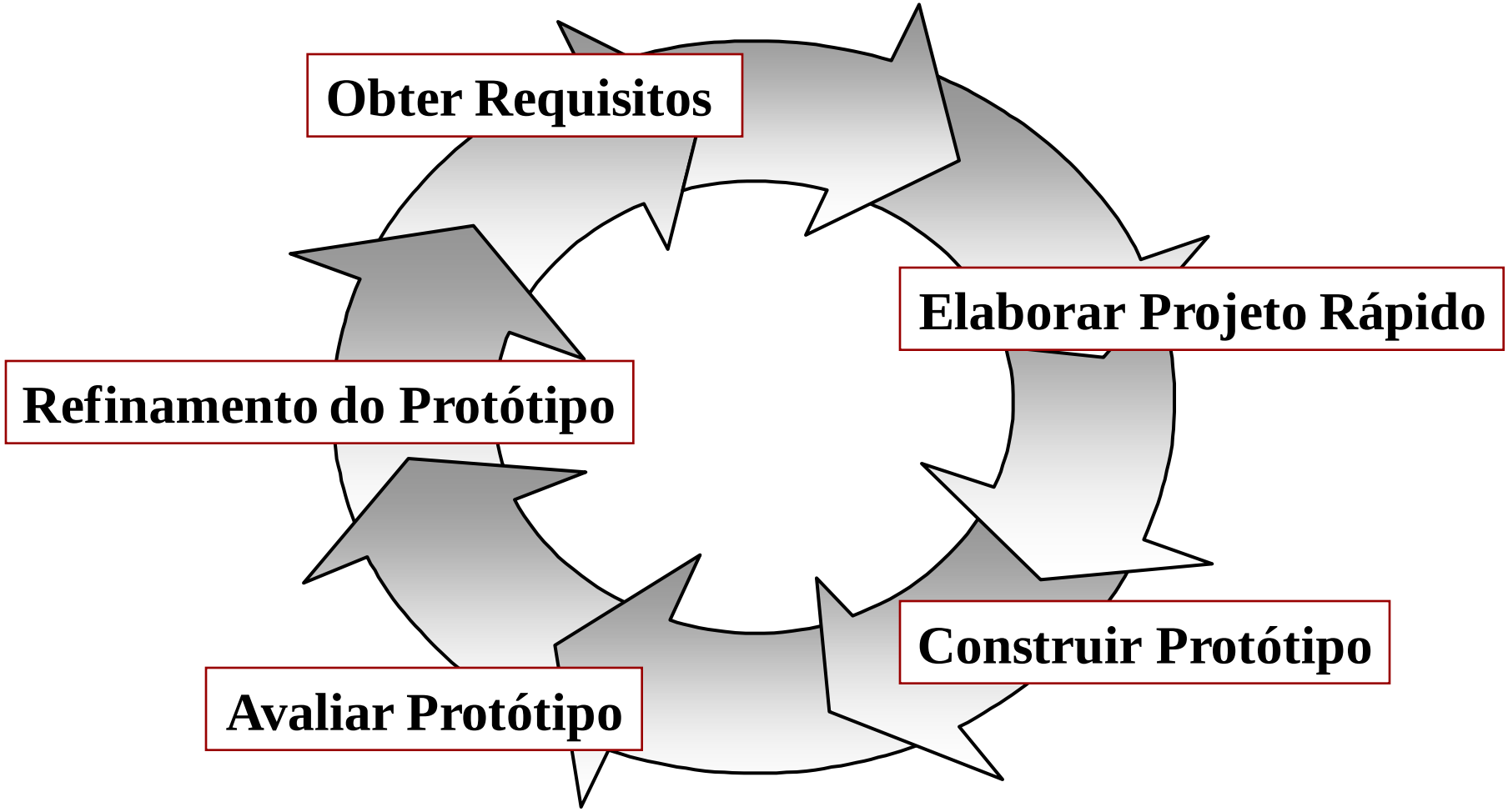
Modelos de Processo de Software

- *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 Espiral***
  - *O Modelo de Montagem de Componentes*
  - *O Modelo de Desenvolvimento Concorrente*
- *Modelos de Métodos Formais*
- *Técnicas de Quarta Geração*

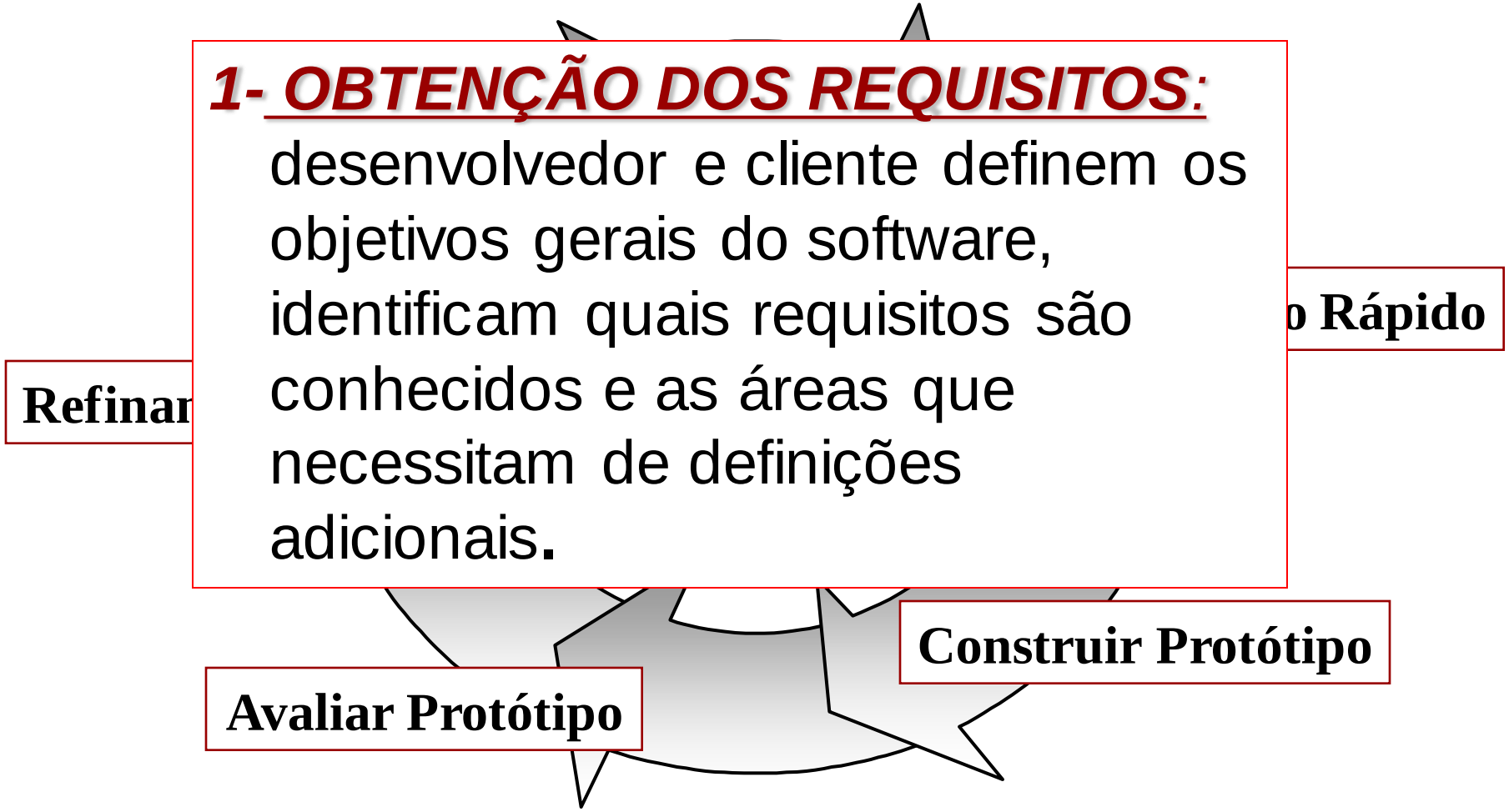
O Modelo de Prototipação

- 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

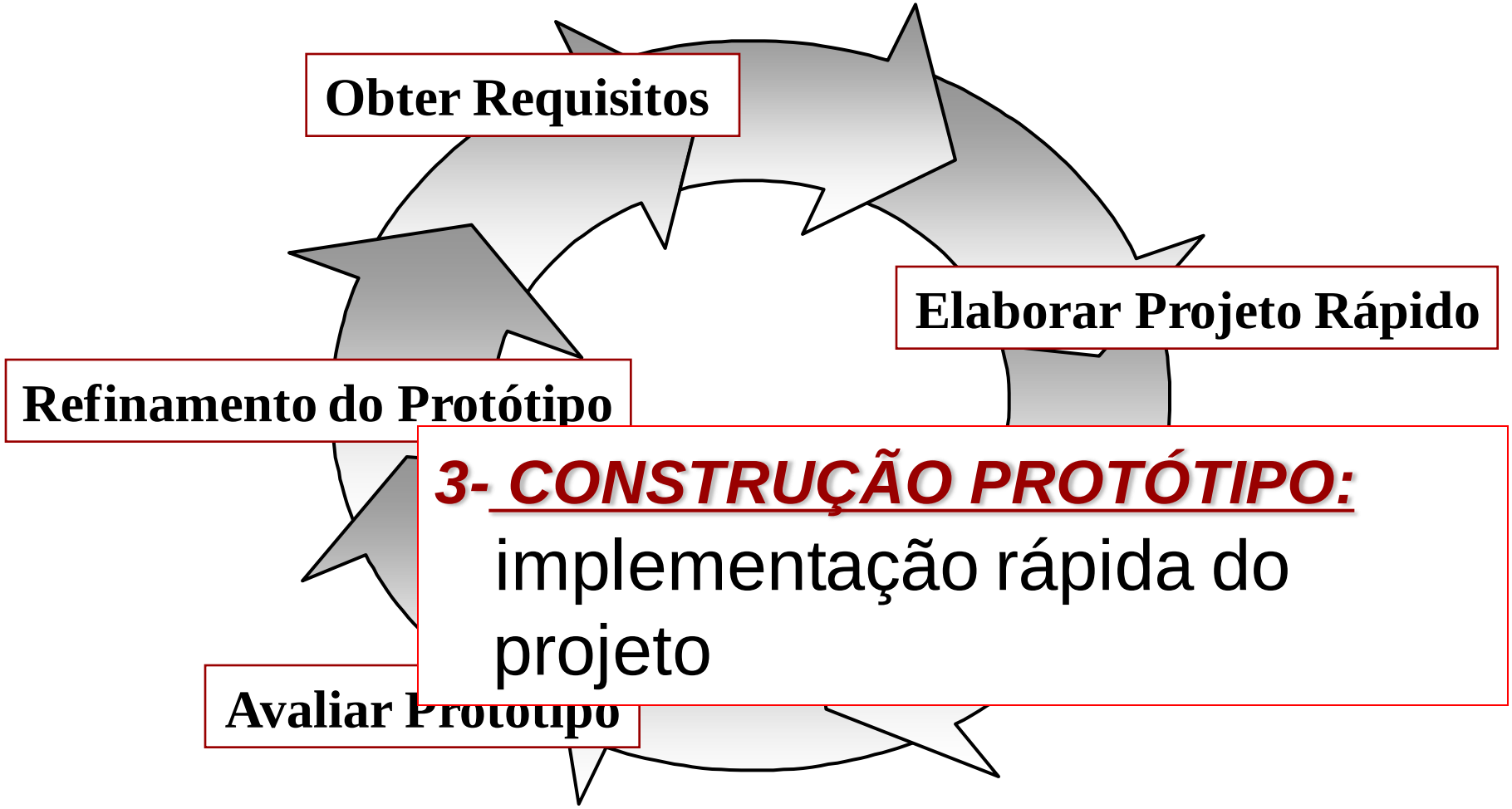


O Paradigma de Prototipação
para obtenção dos requisitos

2- PROJETO RÁPIDO:

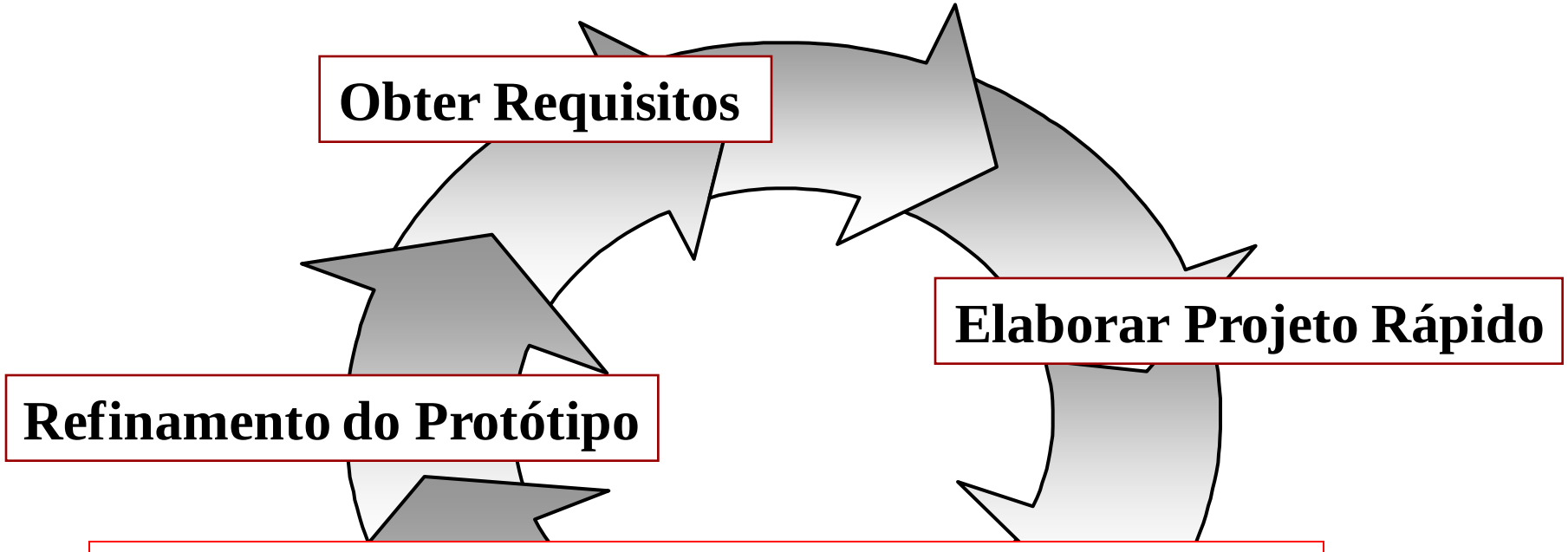
representação dos aspectos do software que são visíveis ao usuário (abordagens de entrada e formatos de saída)

O Paradigma de Prototipação
para obtenção dos requisitos



3- CONSTRUÇÃO PROTÓTIPO:
implementação rápida do projeto

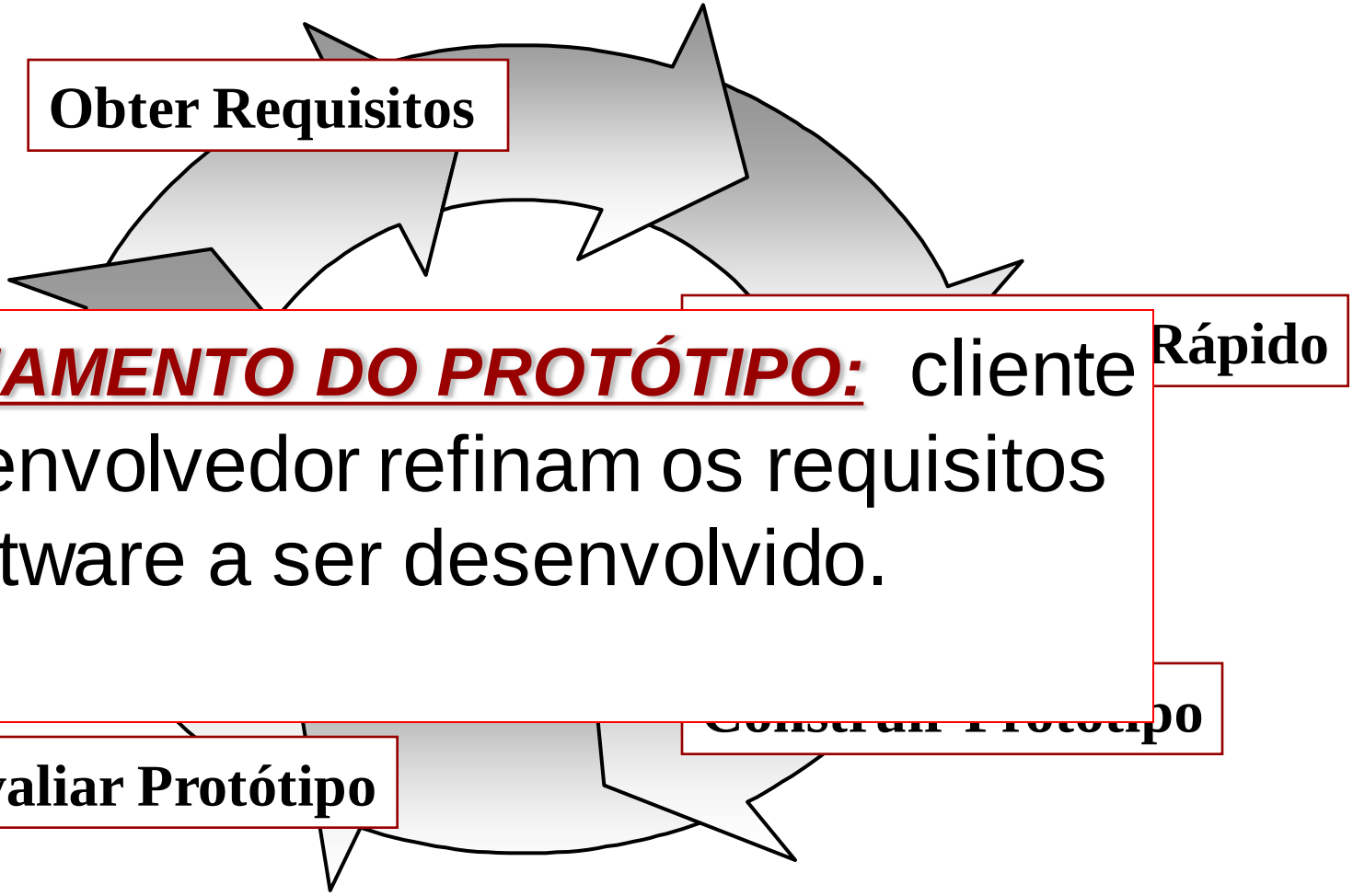
O Paradigma de Prototipação
para obtenção dos requisitos



4- AVALIAÇÃO DO PROTÓTIPO:

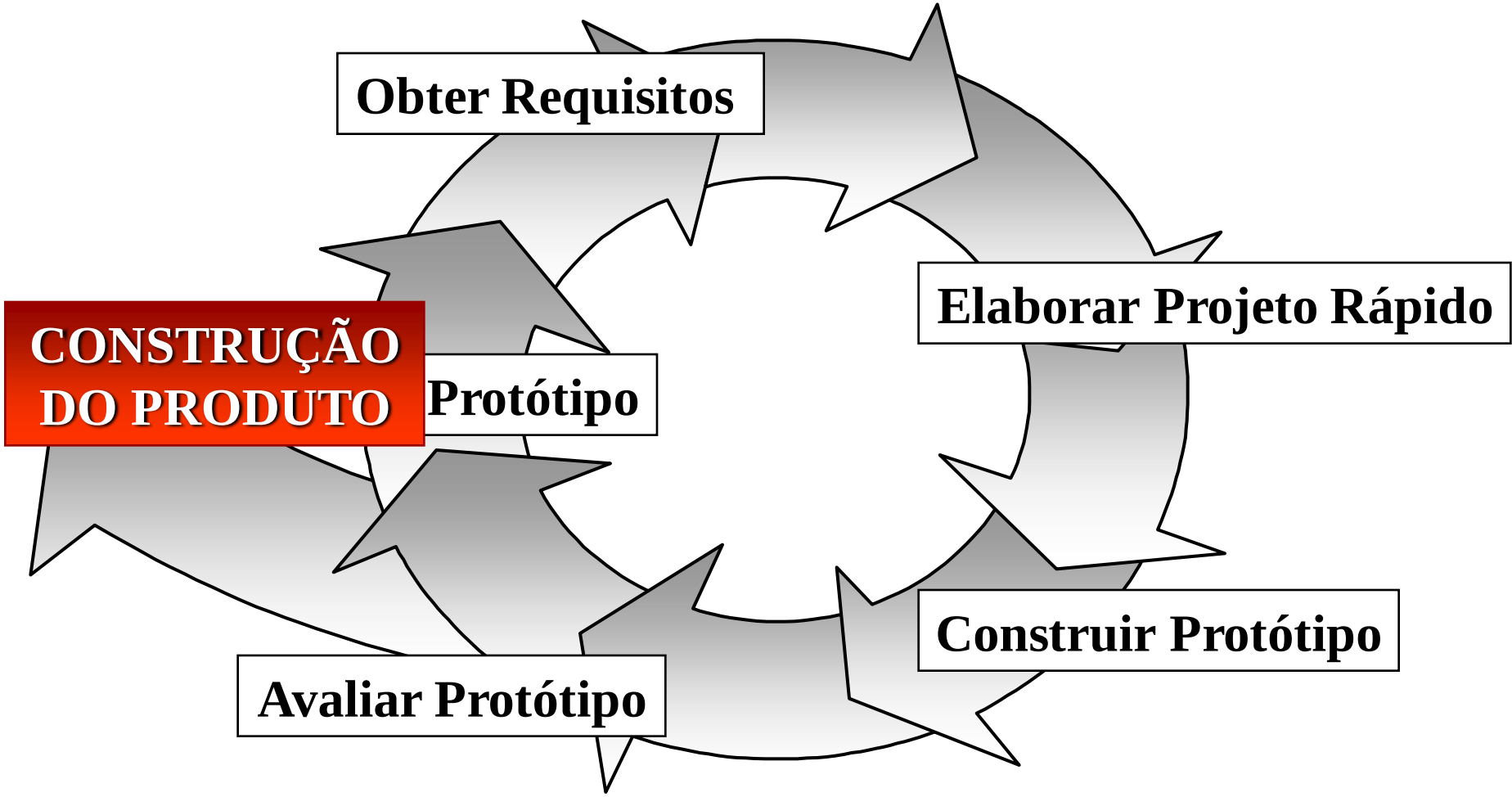
cliente e desenvolvedor avaliam o protótipo

O Paradigma de Prototipação
para obtenção dos requisitos

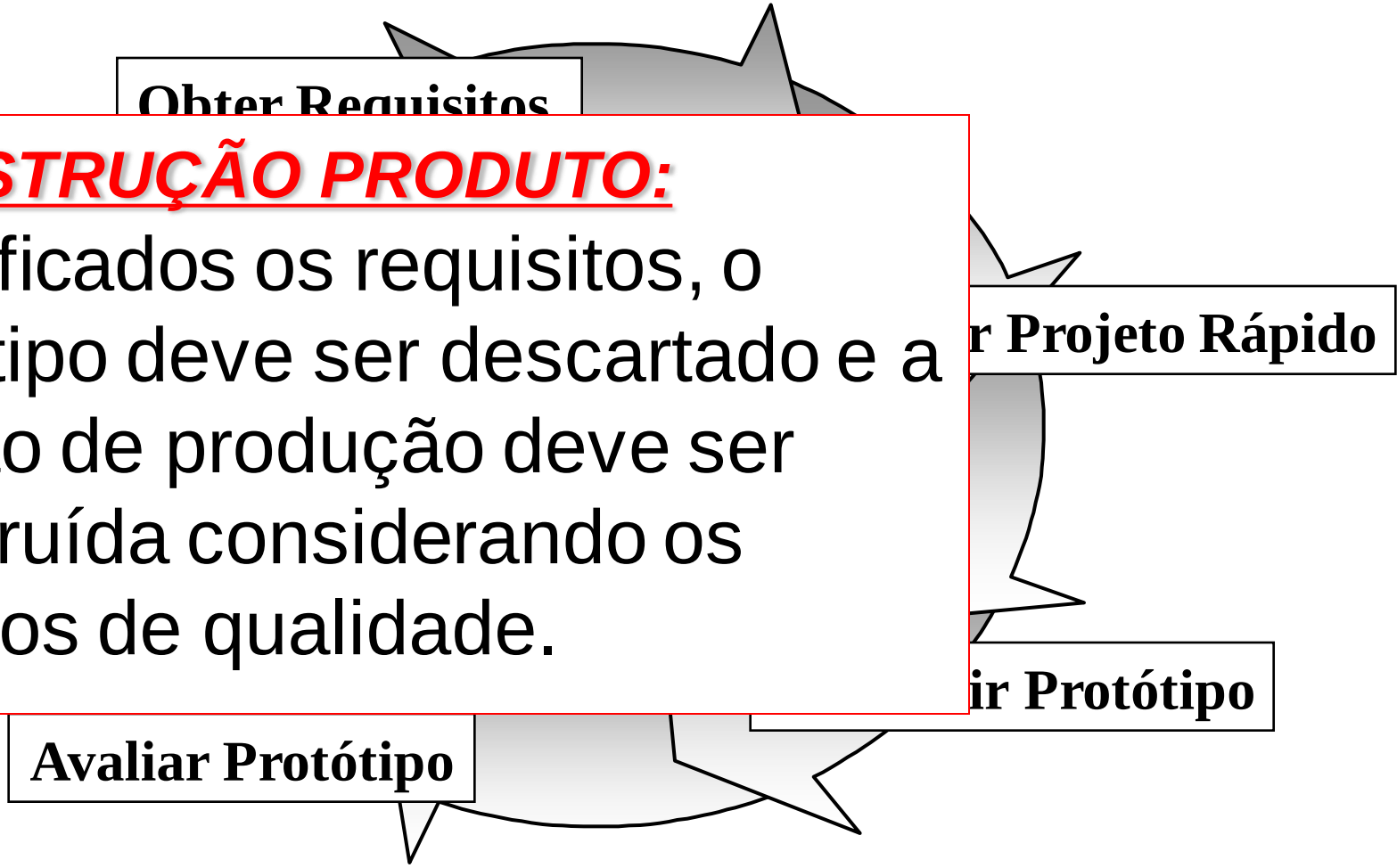


5- REFINAMENTO DO PROTÓTIPO: cliente e desenvolvedor refinam os requisitos do software a ser desenvolvido.

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

Modelos de Processo de Software

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

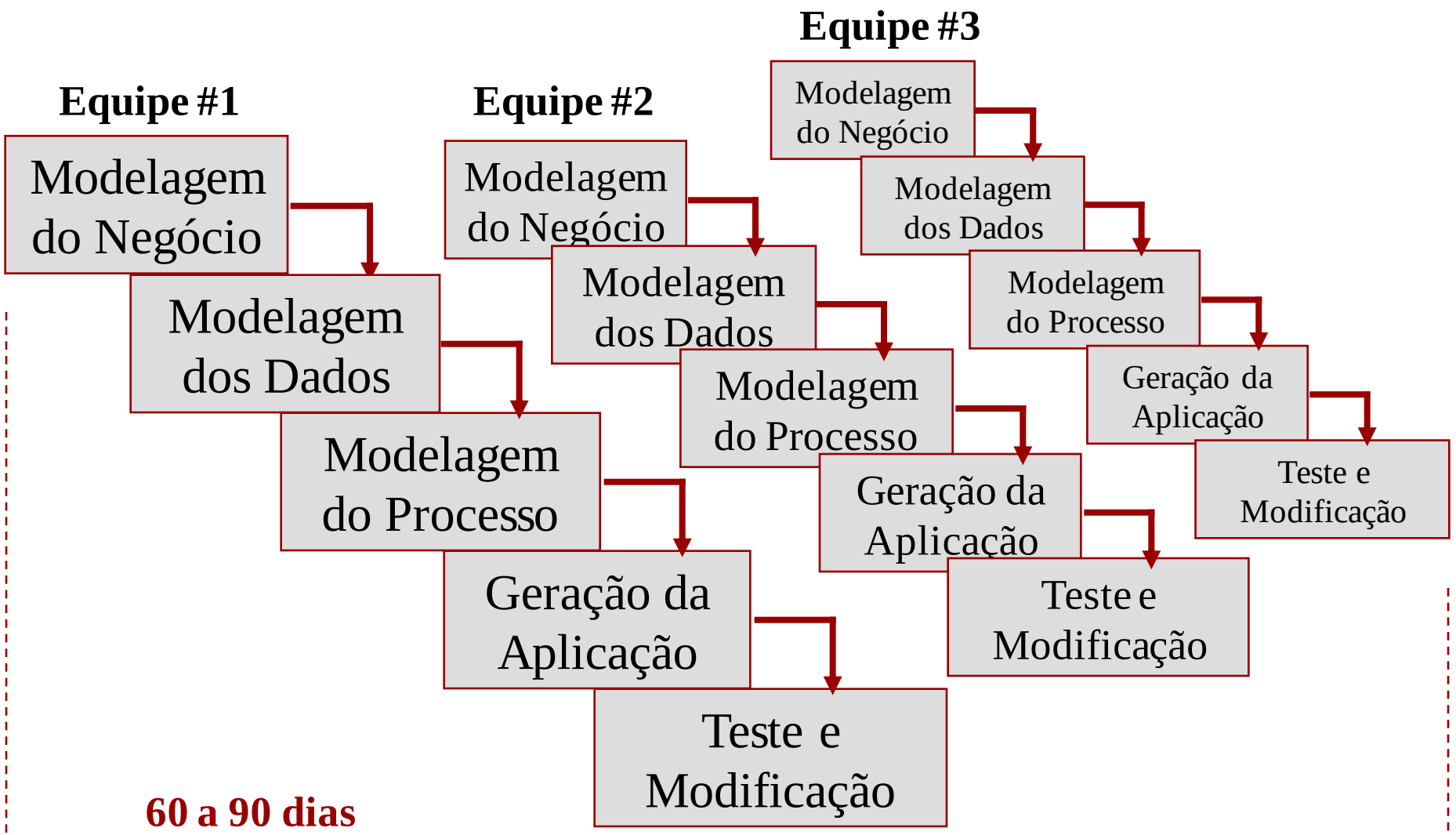
O Modelo RAD

- *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 Paradigma de Prototipação*
- *O Modelo RAD (Rapid Application Development)*
- Modelos Evolutivos de Processo de Software
  - *O Modelo Incremental*
  - *O Modelo Espiral*
  - *O Modelo de Montagem de Componentes*
  - *O Modelo de Desenvolvimento Concorrente*
- *Modelos 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 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

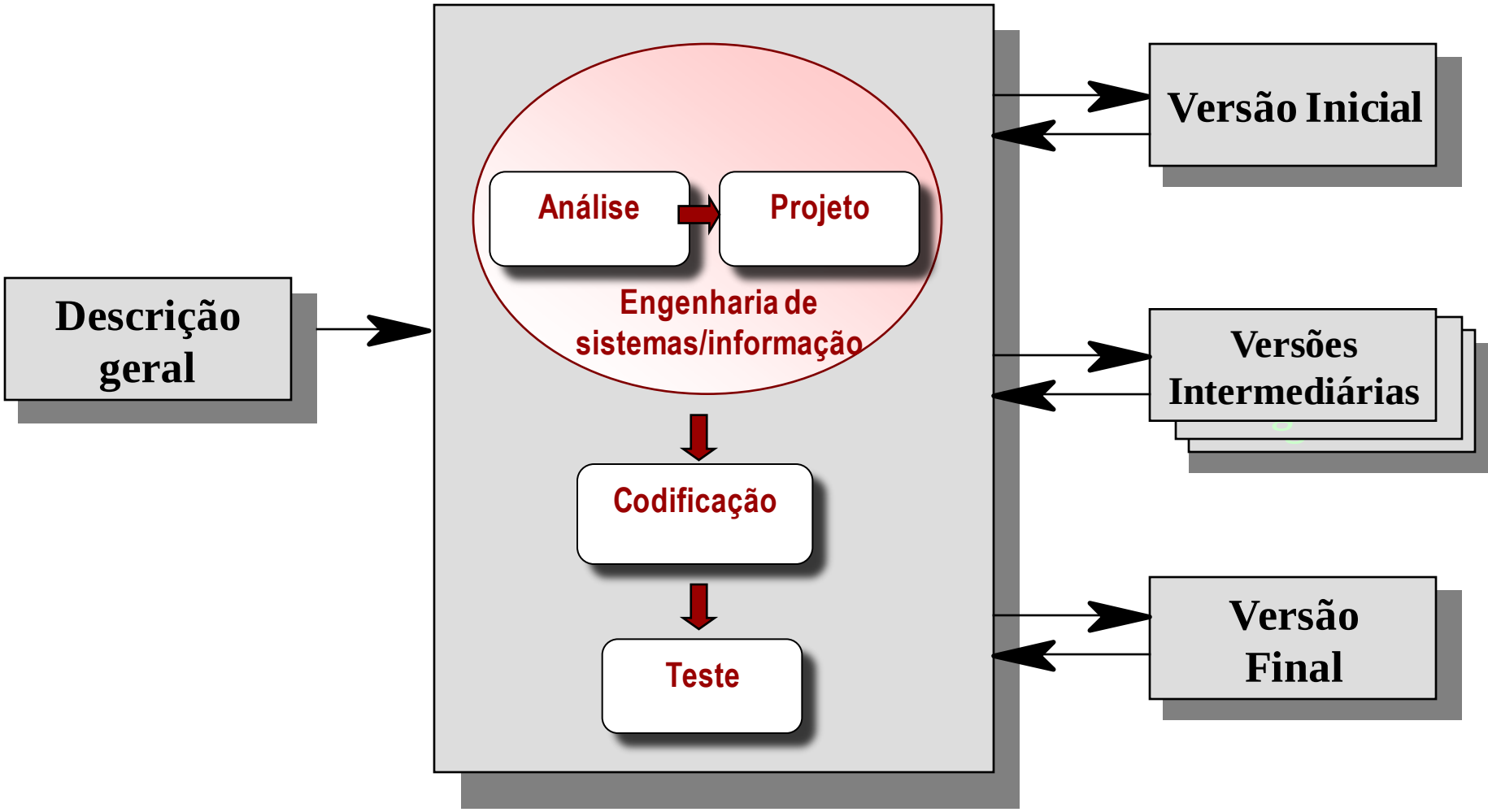
Modelos de Processo de Software

- *O Modelo Sequencial Linear*
  - *também chamado Modelo Cascata*
- *O Paradigma de Prototipação*
- *O Modelo RAD (Rapid Application Development)*
- Modelos Evolutivos de Processo de Software
  - ***O Modelo Incremental***
  - *O Modelo Espiral*
  - *O Modelo de Montagem de Componentes*
  - *O Modelo de Desenvolvimento Concorrente*
- *Modelos 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)

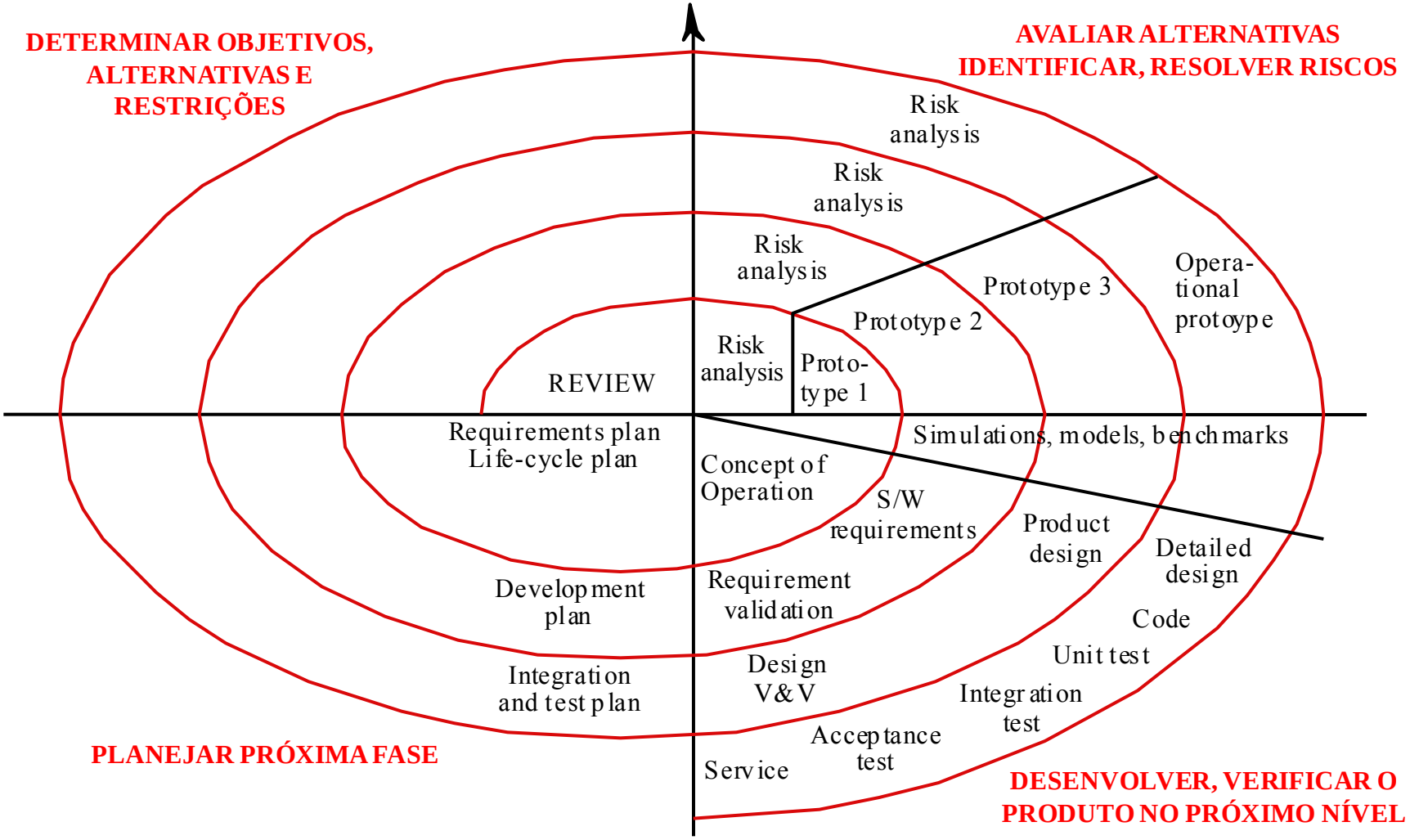
Modelos de Processo de Software

- *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 Espiral***
  - *O Modelo de Montagem de Componentes*
  - *O Modelo de Desenvolvimento Concorrente*
- *Modelos de Métodos Formais*
- *Técnicas de Quarta Geração*

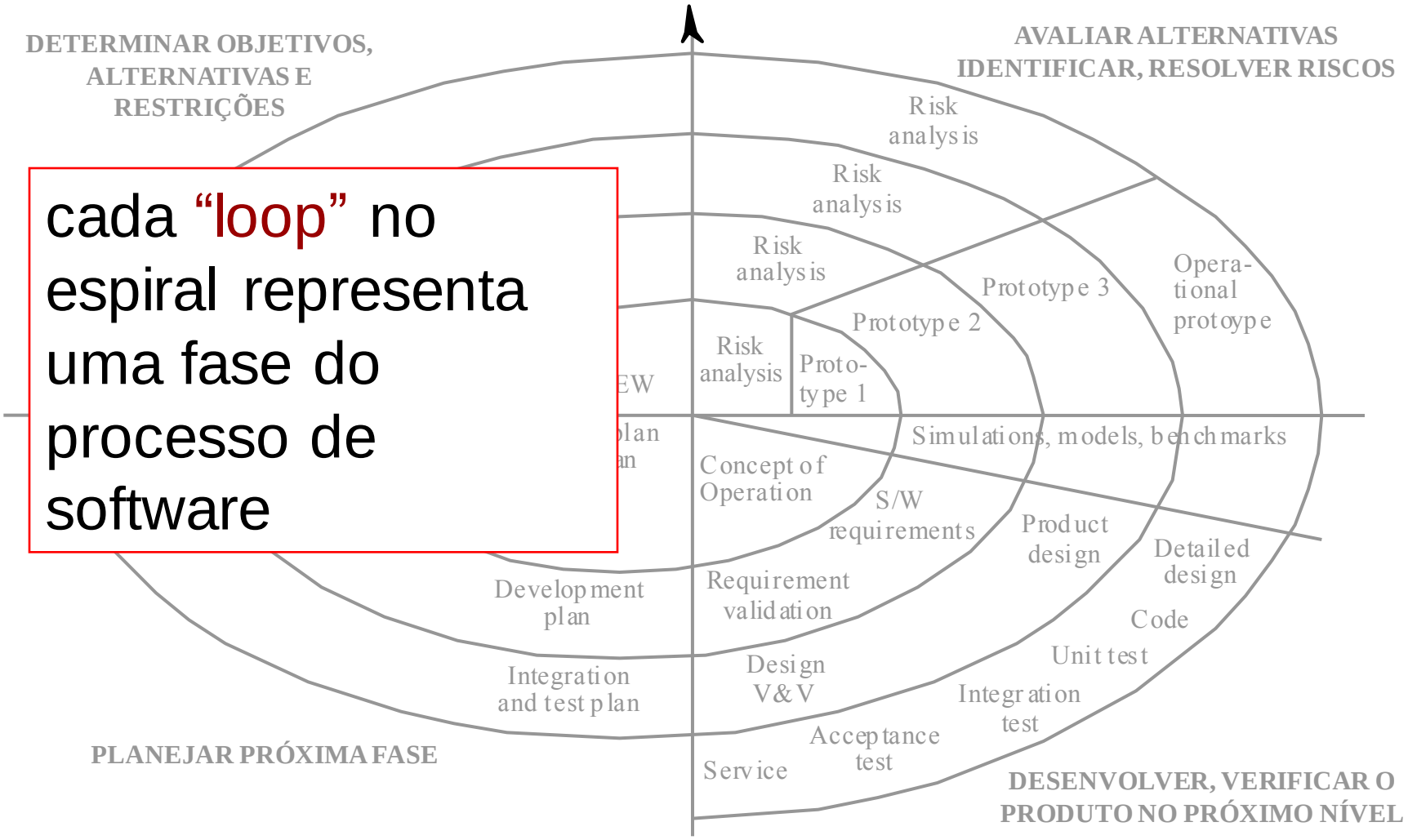
O Modelo Espiral

- 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 (com 4 regiões)



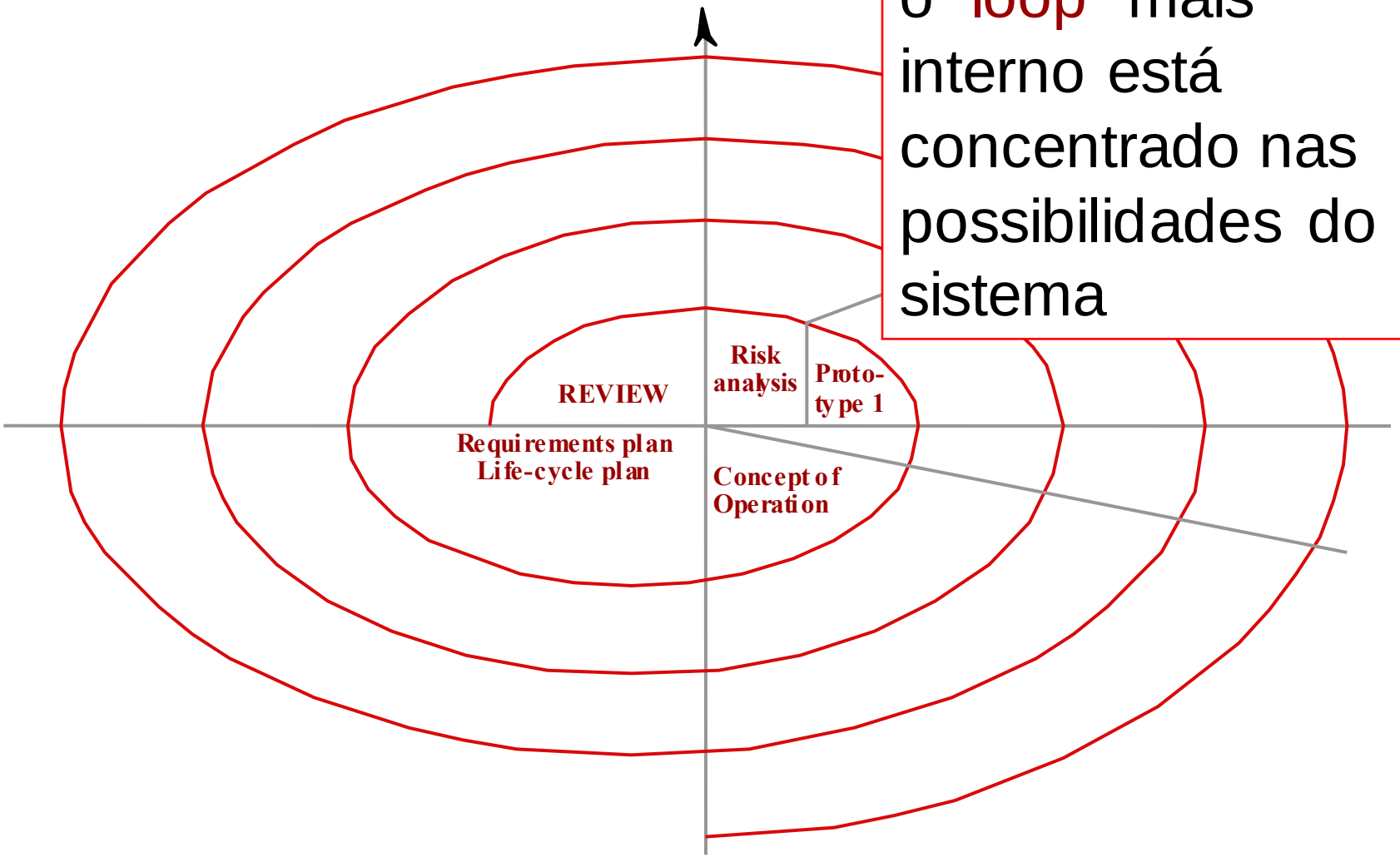
O Modelo Espiral (com 4 regiões)



cada "loop" no espiral representa uma fase do processo de software

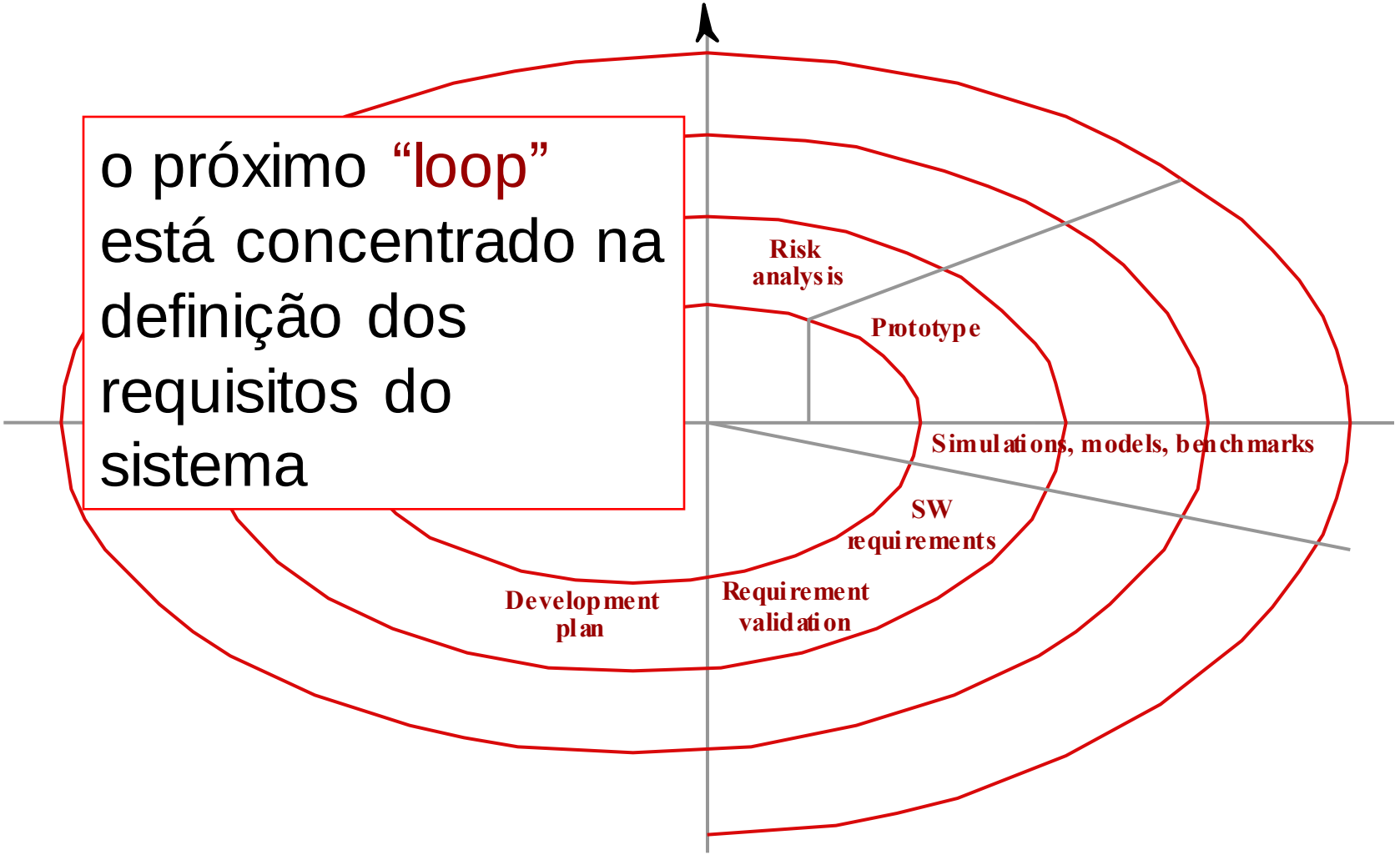
O Modelo Espiral de Processo de Software

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

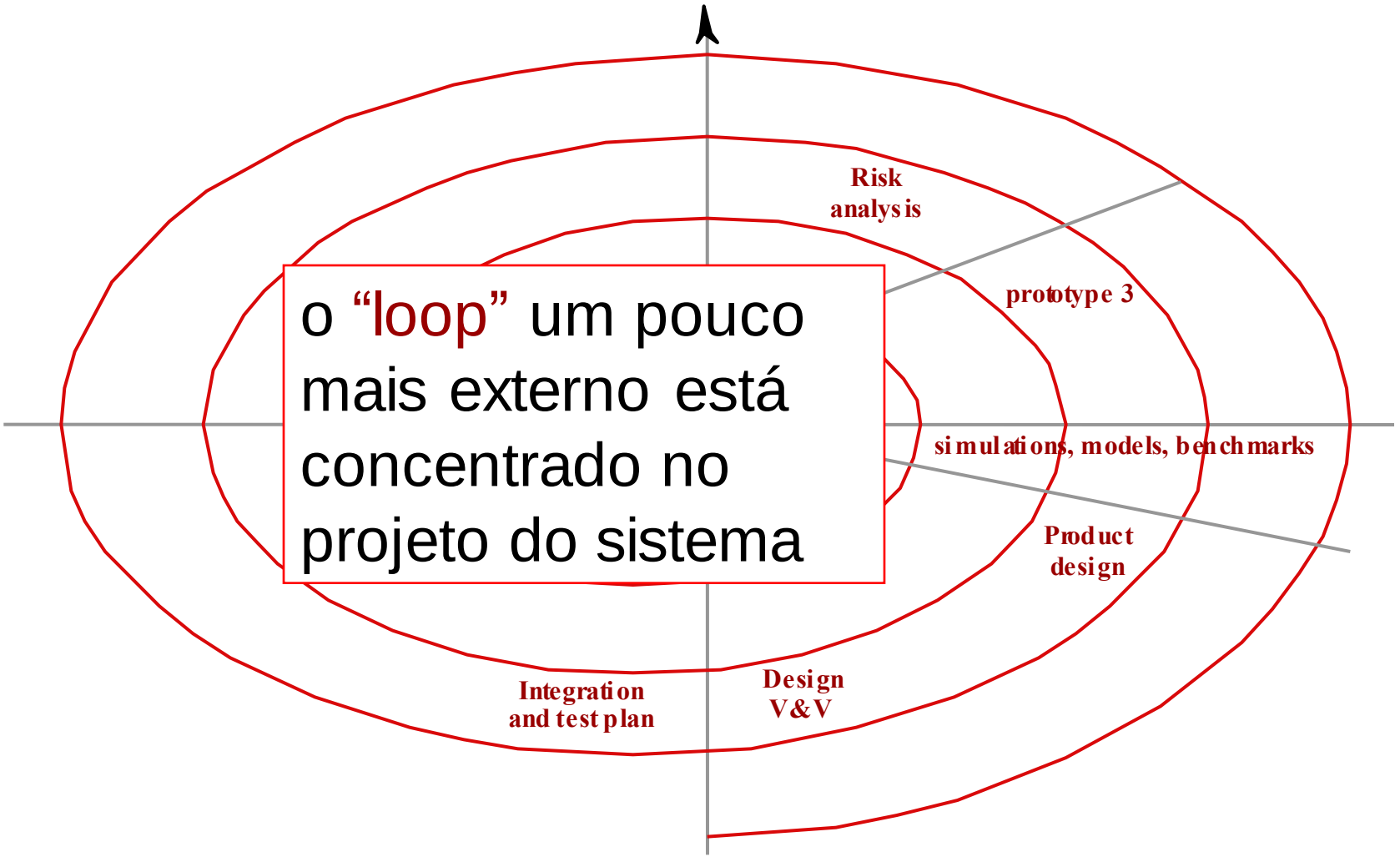


O Modelo Espiral de Processo de Software

o próximo "loop" está concentrado na definição dos requisitos do sistema

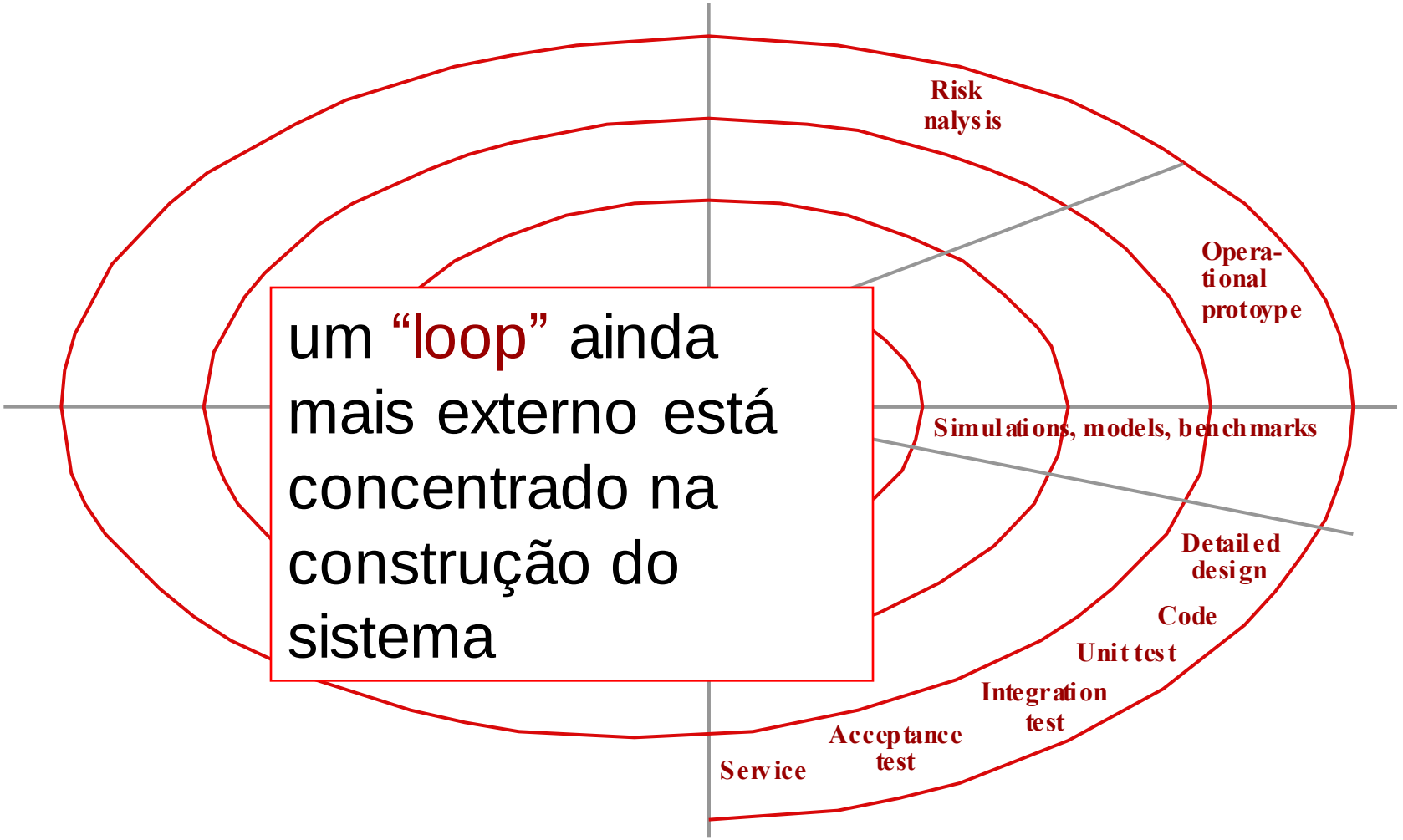


O Modelo Espiral de Processo de Software



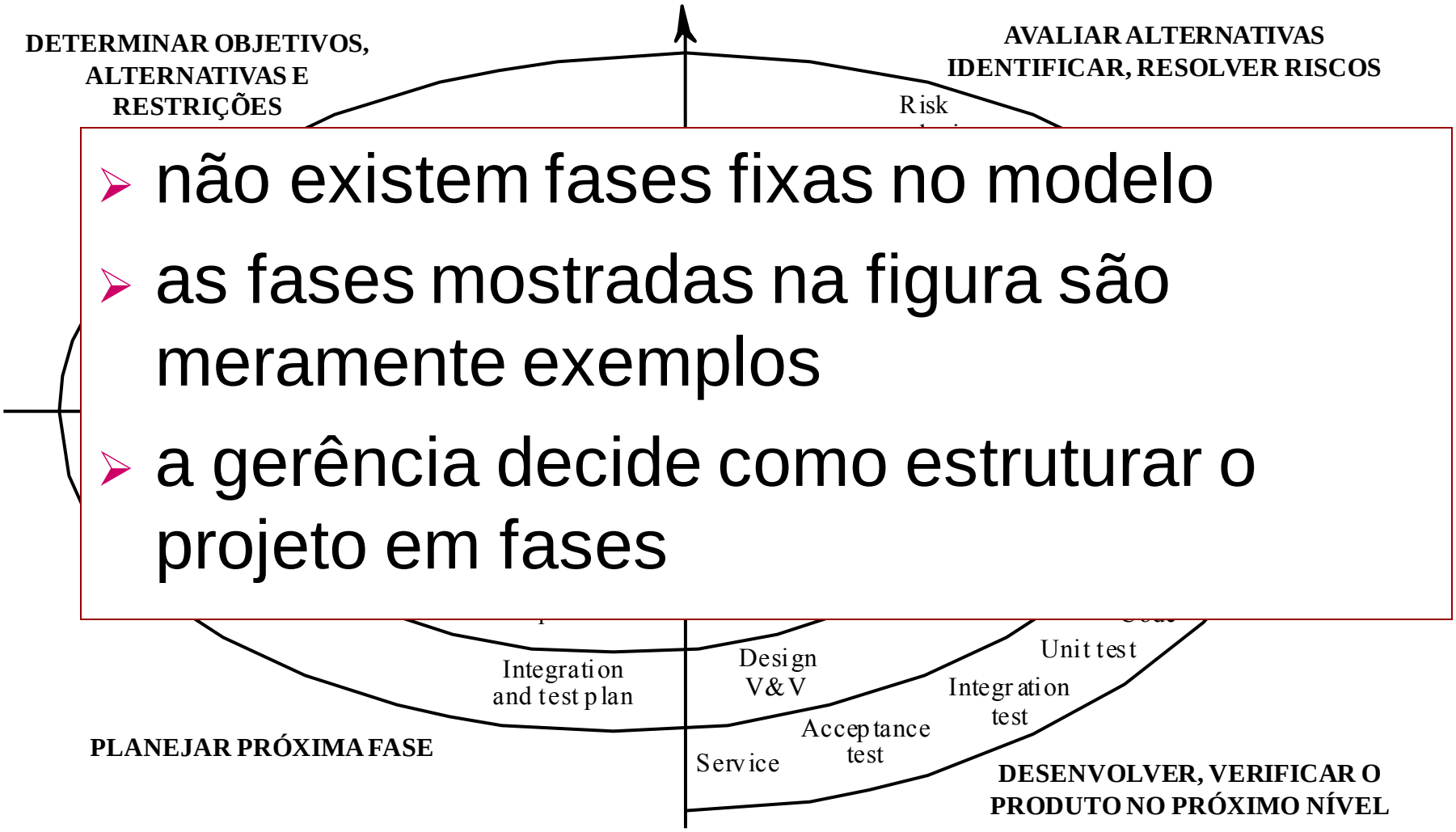
o “loop” um pouco mais externo está concentrado no projeto do sistema

O Modelo Espiral de Processo de Software



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

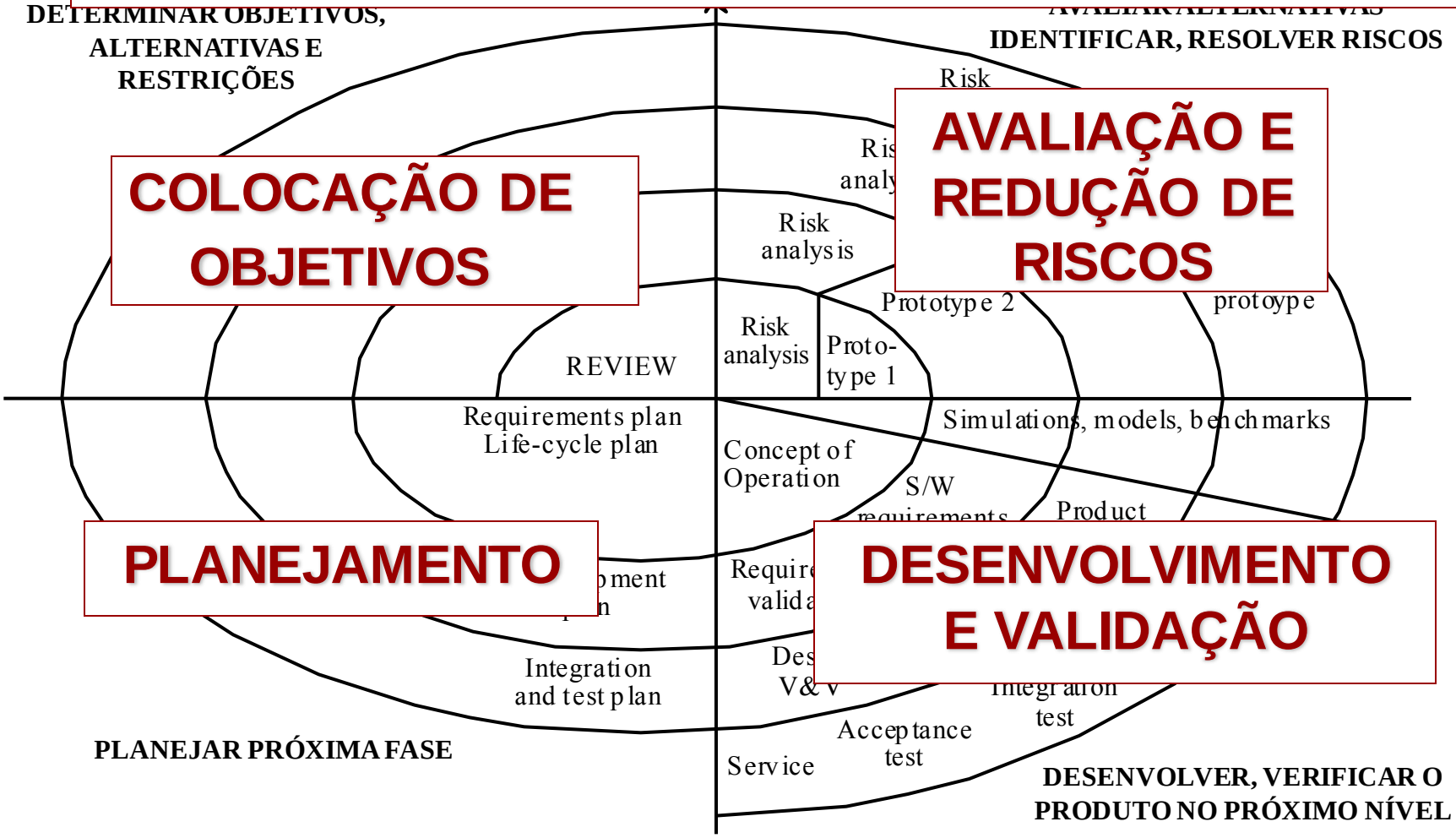
O Modelo Espiral (com 4 regiões)



- 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

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

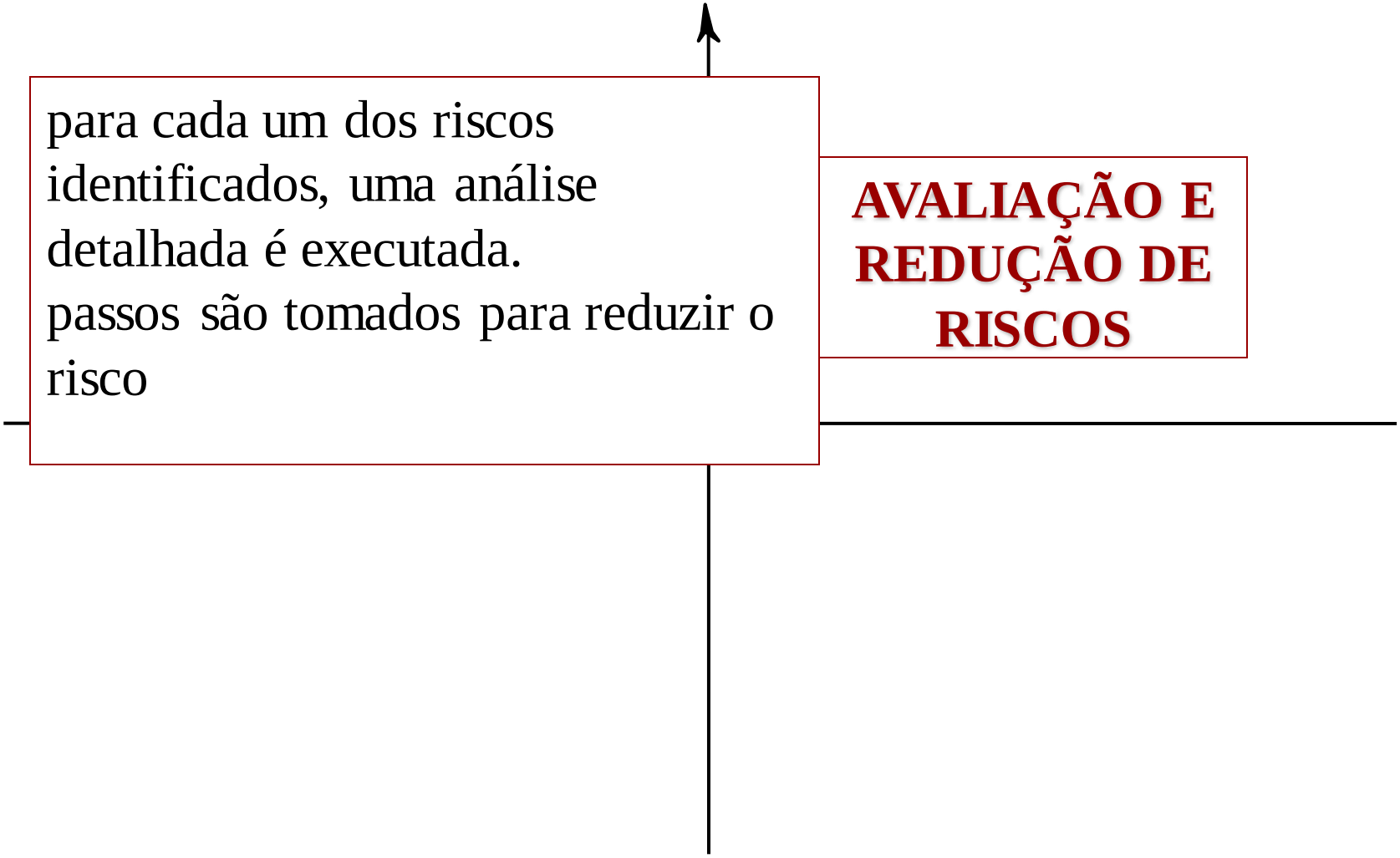


O Modelo Espiral de Processo de Software

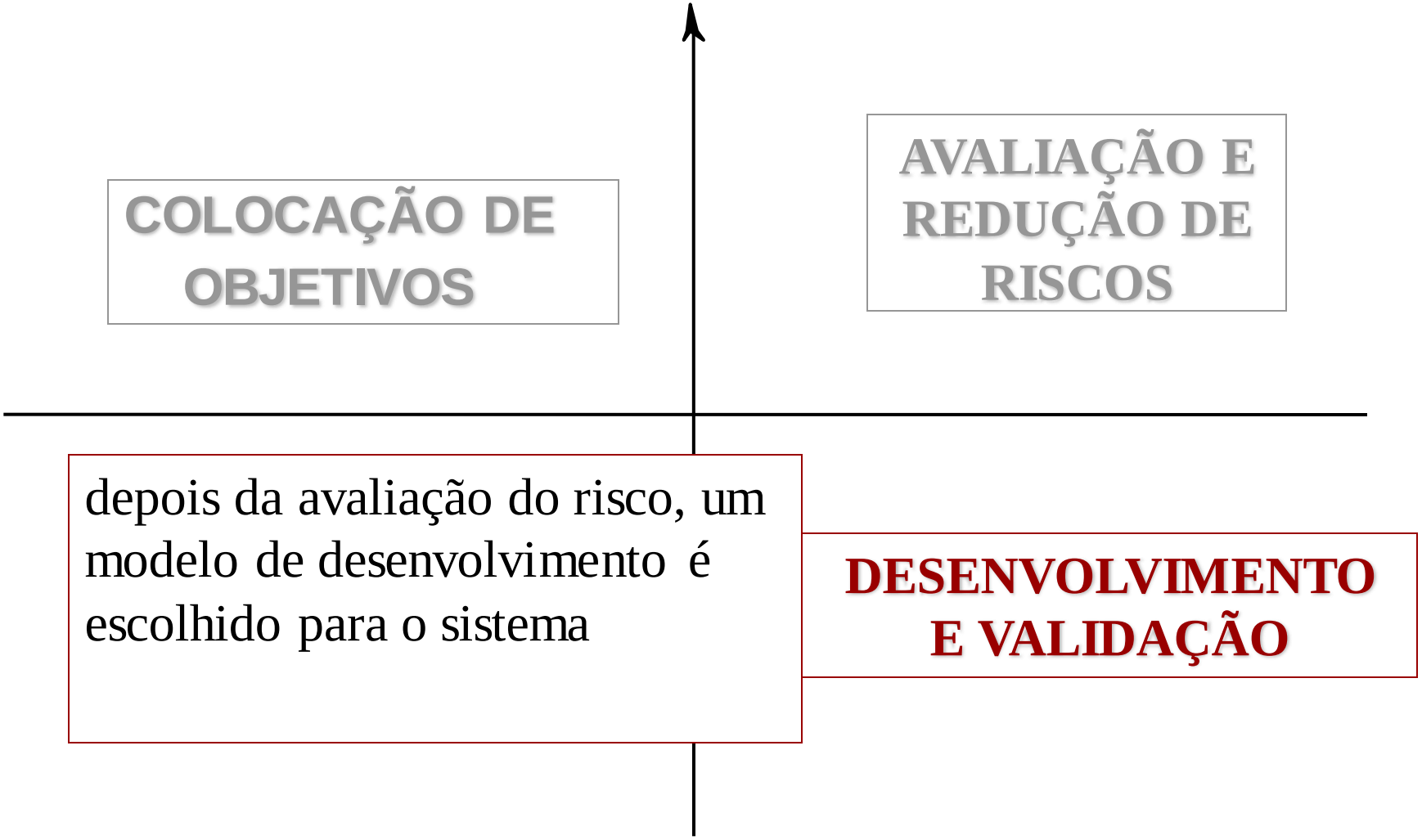
COLOCAÇÃO DE OBJETIVOS

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

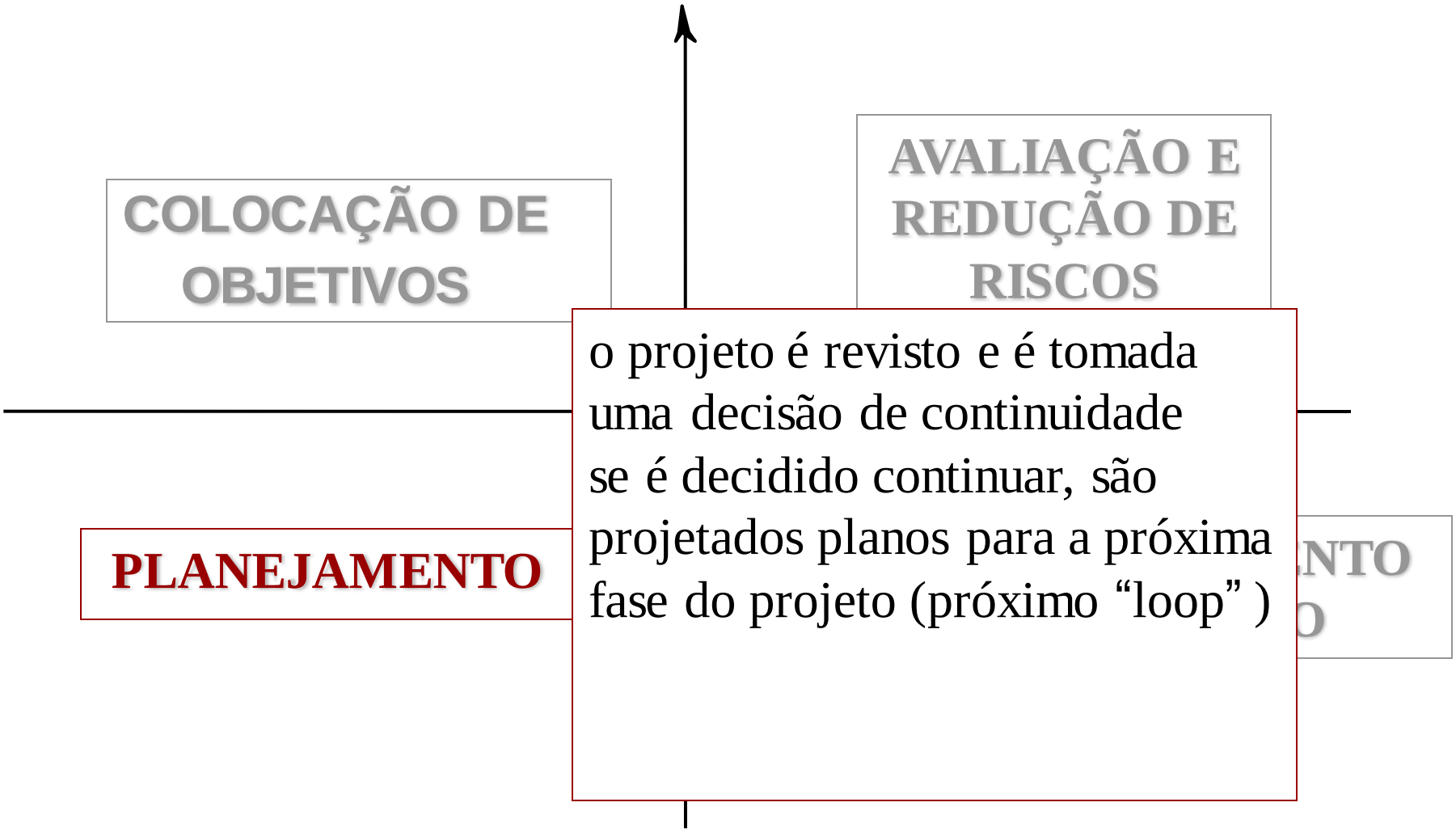
O Modelo Espiral de Processo de Software



O Modelo Espiral de Processo de Software



O Modelo Espiral de Processo de Software



O 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 sistemáti-cos 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

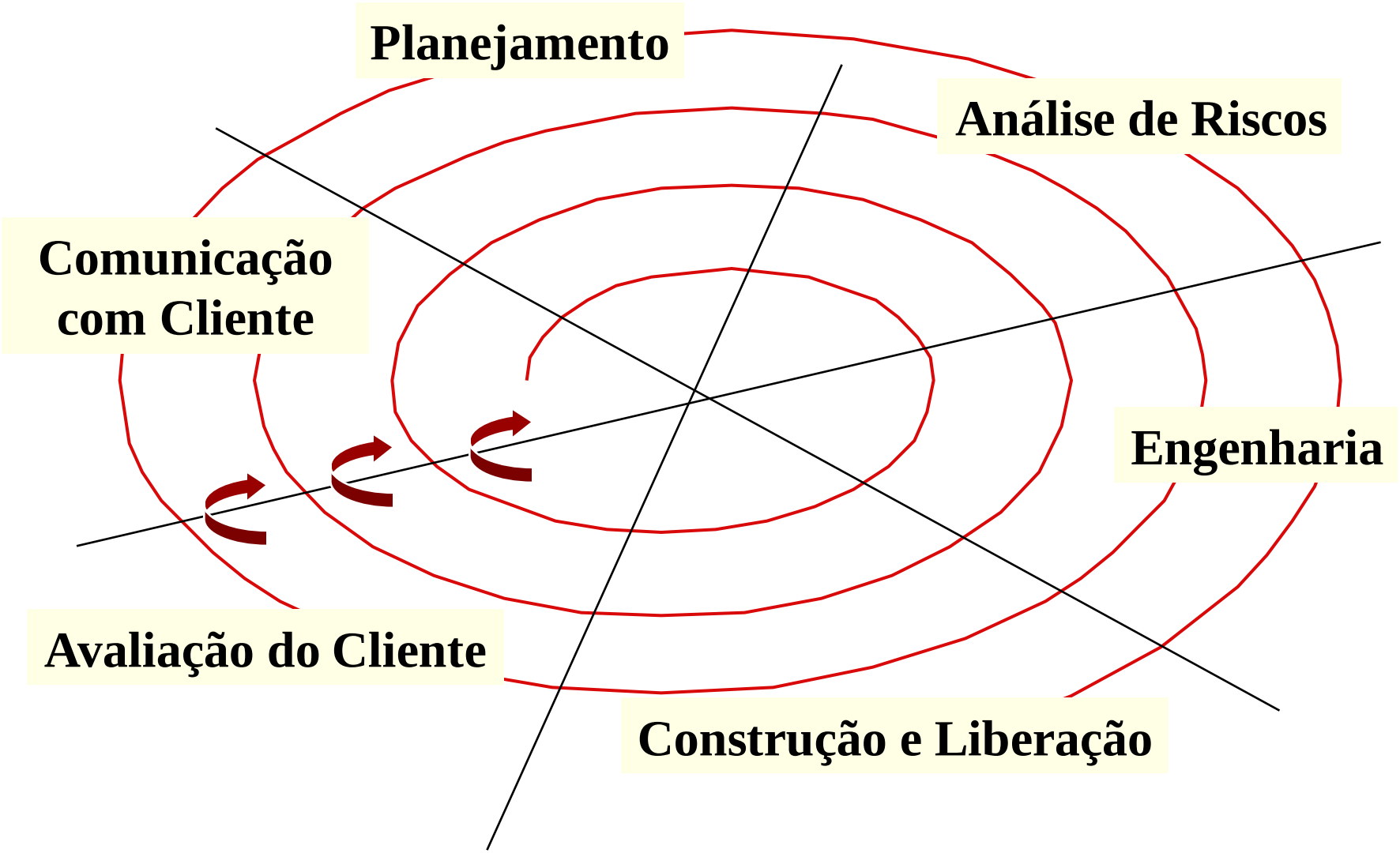
Comentários sobre o Ciclo de Vida em Espiral

- 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

Comentários sobre o Ciclo de Vida em Espiral

- 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

O Modelo Espiral (com 6 regiões)



O Modelo Espiral

- adiciona um novo elemento: a *Análise de Risco*
- usa a Prototipação, em qualquer etapa da evolução do produto, como mecanismo de redução de riscos
- 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.

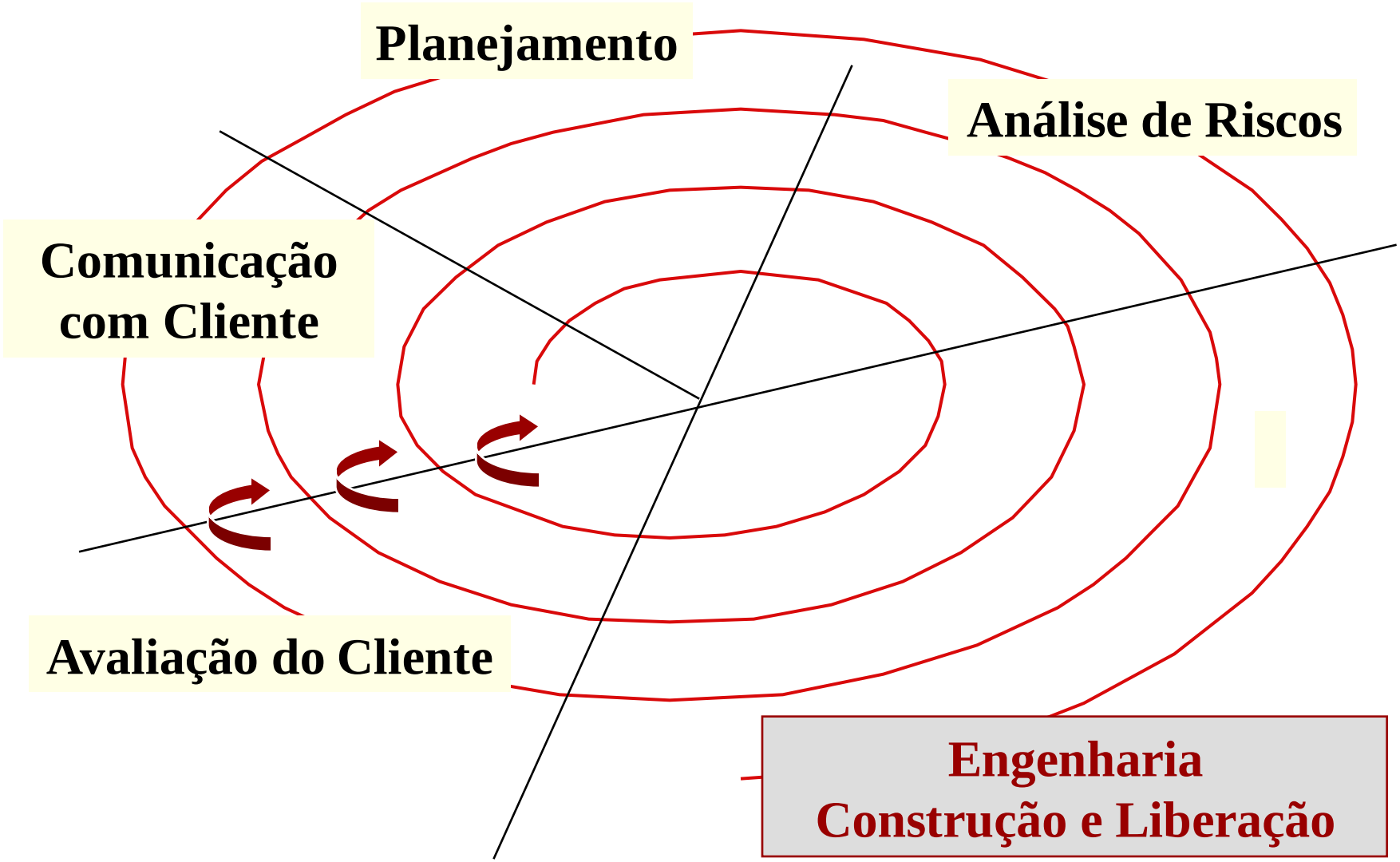
Modelos de Processo de Software

- 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 Espiral
  - **O Modelo de Montagem de Componentes**
  - O Modelo de Desenvolvimento Concorrente
- Modelos de Métodos Formais
- Técnicas de Quarta Geração

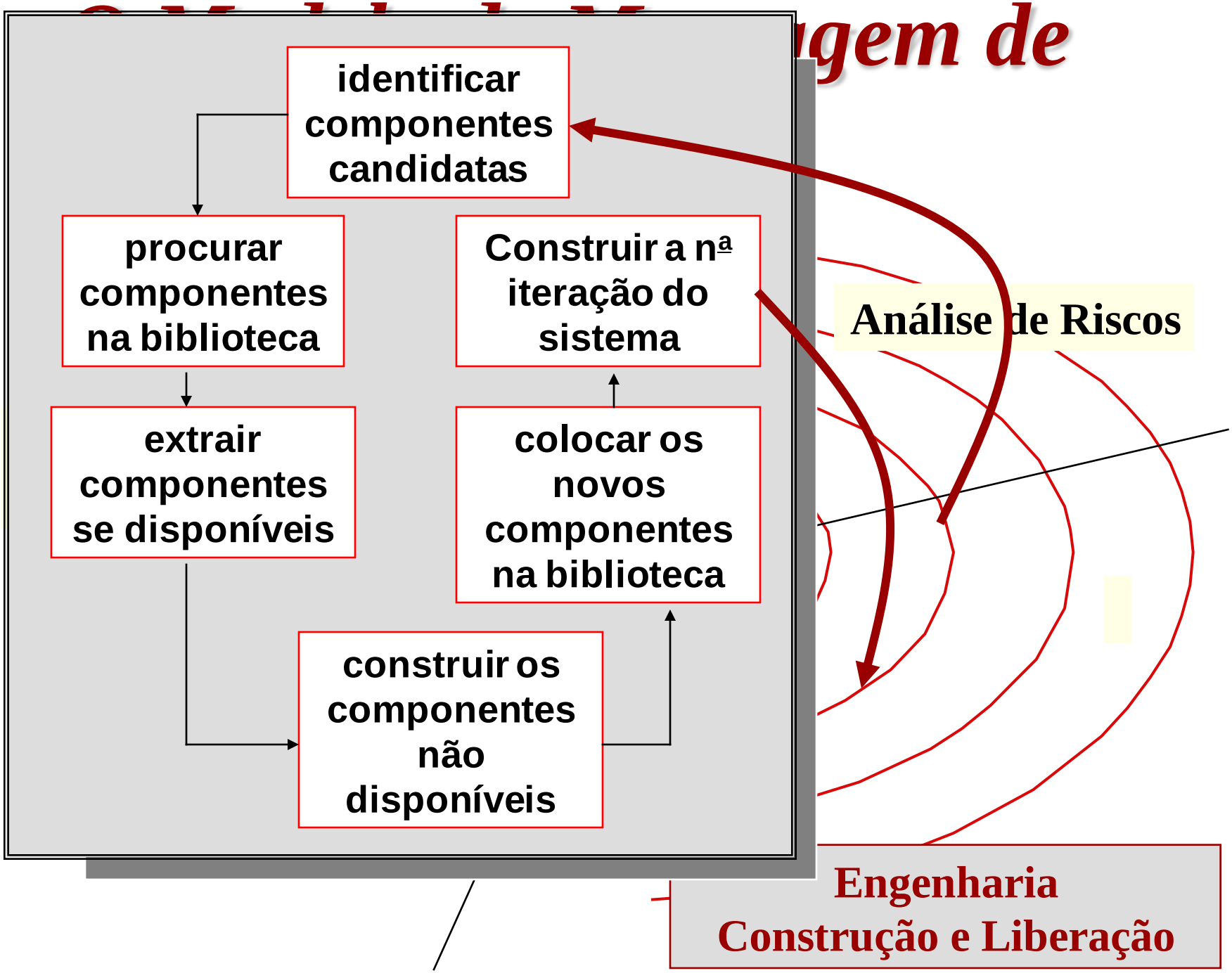
O Modelo de Montagem de Componentes

- 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.

O Modelo de Montagem de Componentes



...gem de


identificar componentes candidatas
procurar componentes na biblioteca
extrair componentes se disponíveis
construir os componentes não disponíveis
colocar os novos componentes na biblioteca
Construir a nª iteração do sistema
Análise de Riscos
Engenharia
Construção e Liberação

O 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

- 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 Espiral
  - O Modelo de Montagem de Componentes
  - O Modelo de Desenvolvimento Concorrente
- Modelos de Métodos Formais
- Técnicas de Quarta Geração

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 e
  - ⇨o código fonte seja gerado automaticamente a partir dessas especificações

Ferramentas do Ambiente das 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
  - o cliente pode ser incapaz de especificar as informações de um modo que uma ferramenta 4GL possa consumir
  - as 4GLs atuais não são sofisticadas suficientemente para acomodar a verdadeira "linguagem natural"

Técnicas de 4ª Geração

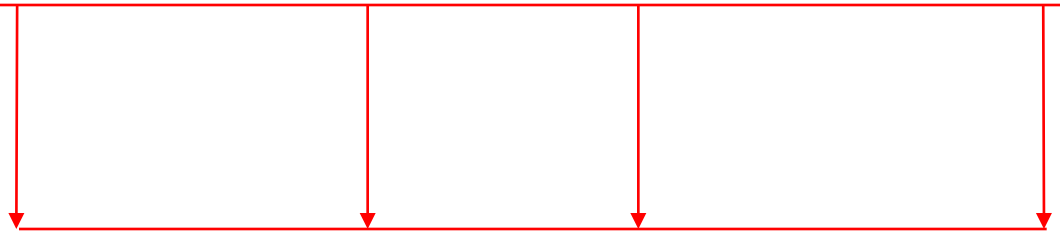
- **ESTRATÉGIA DO "PROJETO":**
  - Para pequenas aplicações é possível mover-se do passo de Obtenção dos Requisitos para o passo de Implementação usando uma linguagem de quarta geração
  - Para grandes projetos é necessário desenvolver uma estratégia de projeto. De outro modo ocorrerão os mesmos problemas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

Técnicas de 4ª Geração

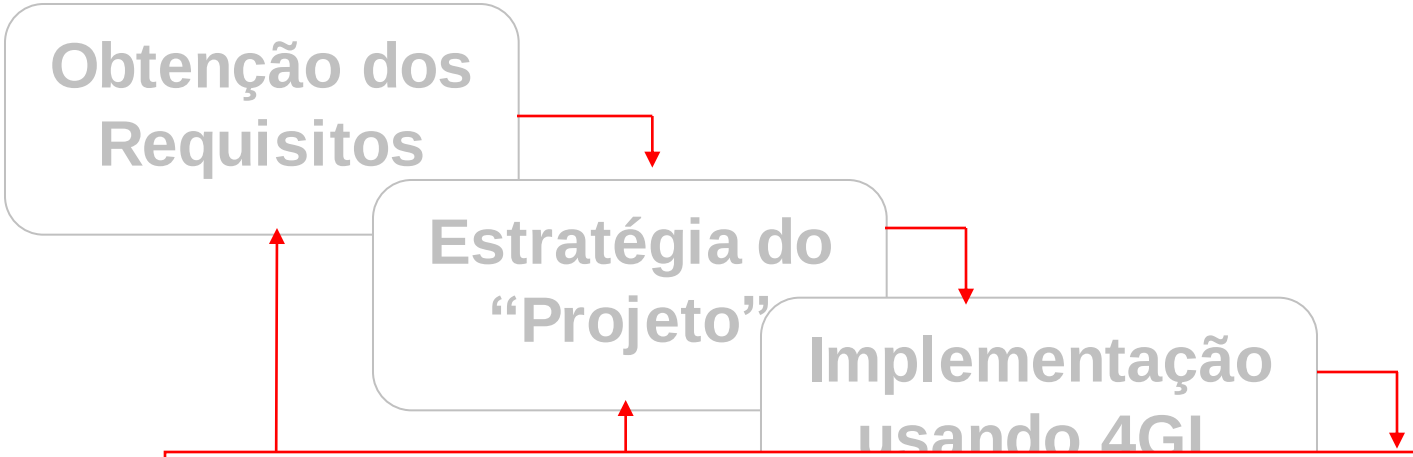
Obtenção dos Requisitos

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



Técnicas de 4ª Geração



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.

Comentários sobre as Técnicas de 4ª Geração

- PROPONENTES:
  - redução dramática no tempo de desenvolvimento do software (aumento de produtividade)
- OPONENTES:
  - 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

Para escolha de um Modelo de Processo de Software:

- **natureza do projeto e da aplicação**
- **métodos e ferramentas a serem usados**
- **controles e produtos que precisam ser entregues**