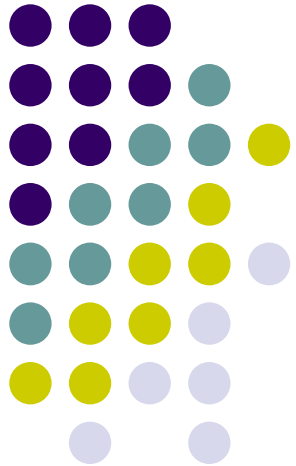
Engenharia de Software 1 (ES1) (1001530)

Aula 2:

Modelos de Processo de Desenvolvimento de Software



Prof. Fabiano Cutigi Ferrari

2º semestre de 2022

Recados Iniciais

- Mantenham seus e-mails atualizados no Moodle.

Roteiro

- Fases Genéricas de um Processo de Software.
- Modelos de Processo de Software
  - Modelo Cascata
  - Modelo RAD
  - Modelos Evolutivos
  - Métodos Ágeis

Perguntas Iniciais

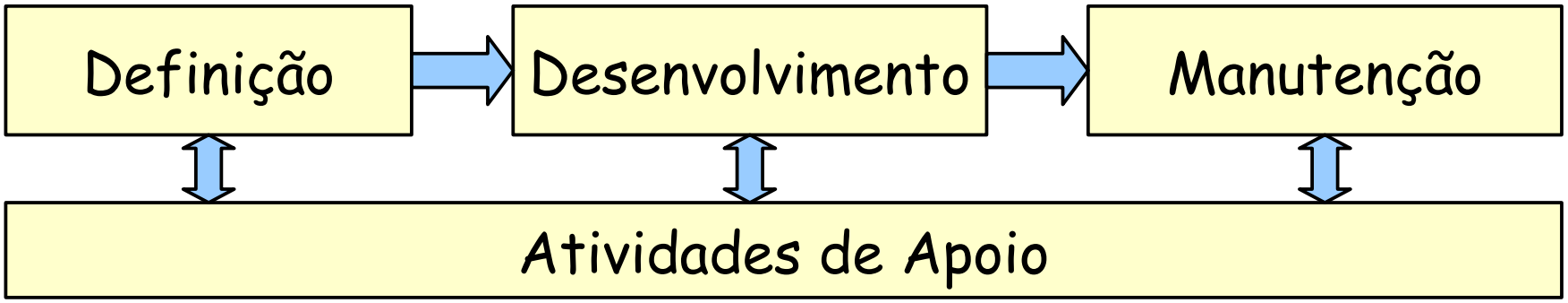
- Como vocês desenvolvem software?
- Quais atividades realizam?
- Em que ordem executam as atividades?

Roteiro

- **Fases Genéricas de um Processo de Software.**
- Modelos de Processo de Software
  - Modelo Cascata
  - Modelo RAD
  - Modelos Evolutivos
  - Métodos Ágeis

Fases Genéricas dos Modelos de Processo de Software

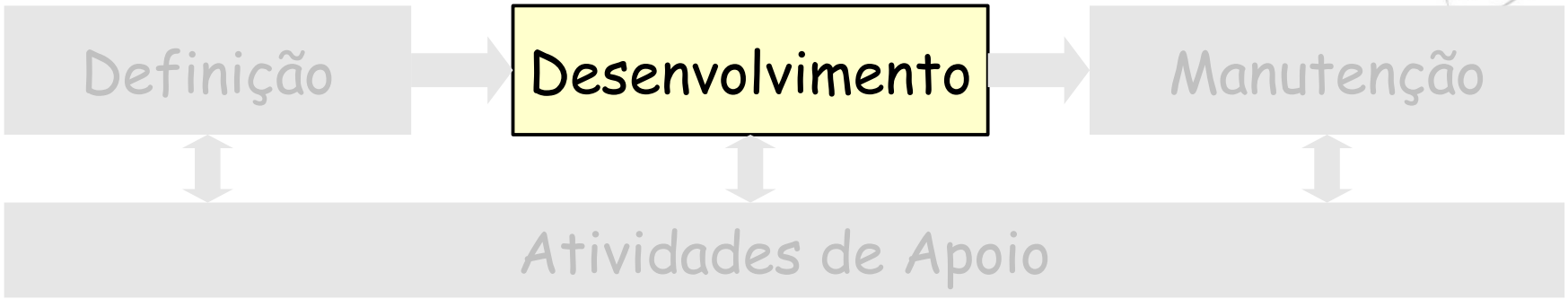
- Independentemente da natureza do projeto e aplicação os modelos de processo de software possuem:
  - fase de definição
  - fase de desenvolvimento
  - fase de manutenção
  - atividades de apoio



Fases Genéricas dos Modelos de Processo de Software

- **Focaliza "o que" de software será desenvolvido.**
- Focaliza "como" o software será desenvolvido.
- Focaliza as "mudanças" pelas quais o software passará depois de entregue.
- Complementam as três fases genérias, sendo realizadas durante toda a engenharia do software.

Fases Genéricas dos Modelos de Processo de Software

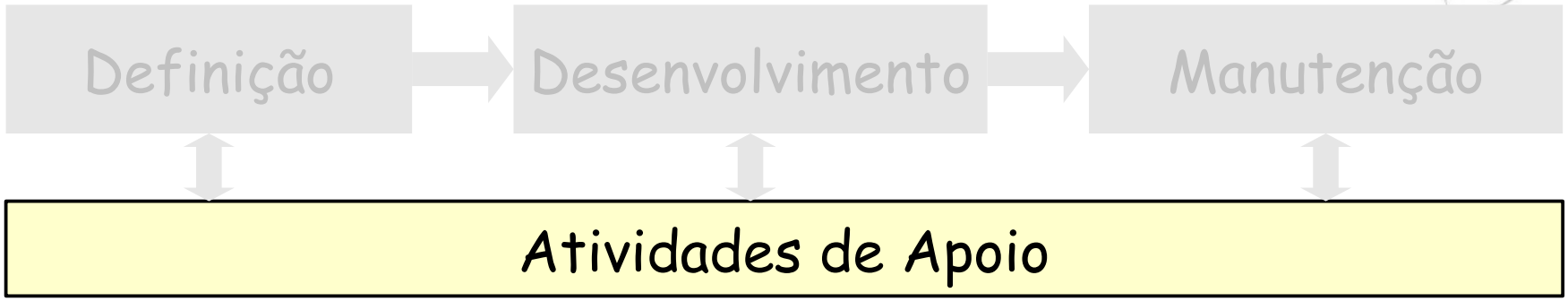


- Focaliza "o que" de software será desenvolvido.
- **Focaliza "como" o software será desenvolvido.**
- Focaliza as "mudanças" pelas quais o software passará depois de entregue.
- Complementam as três fases genérias, sendo realizadas durante toda a engenharia do software.

Fases Genéricas dos Modelos de Processo de Software

- Focaliza "o que" de software será desenvolvido.
- Focaliza "como" o software será desenvolvido.
- **Focaliza as "mudanças" pelas quais o software passará depois de entregue.**
- Complementam as três fases genérias, sendo realizadas durante toda a engenharia do software.

Fases Genéricas dos Modelos de Processo de Software



- Focaliza "o que" de software será desenvolvido.
- Focaliza "como" o software será desenvolvido.
- Focaliza as "mudanças" pelas quais o software passará depois de entregue.
- **Complementam as três fases genérias, sendo realizadas durante toda a engenharia do software.**

Roteiro

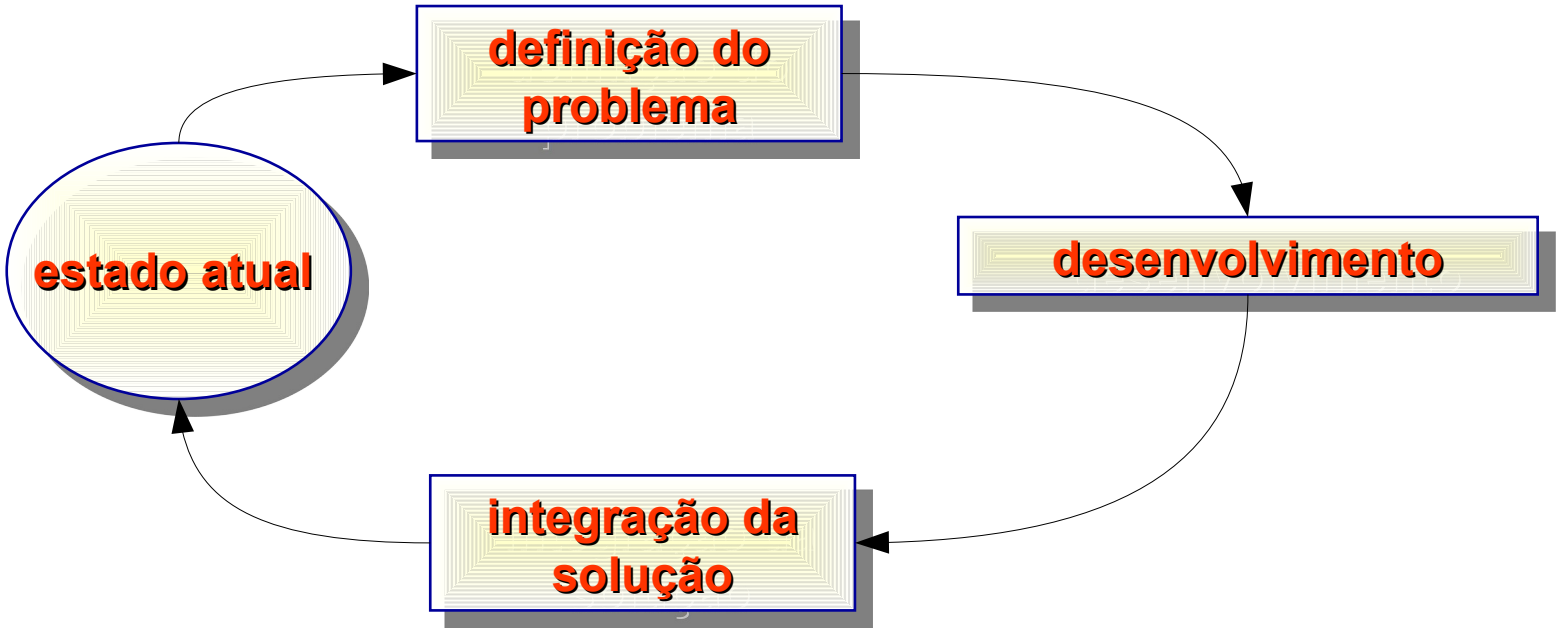
- Fases Genéricas de um Processo de Software.
- **Modelos de Processo de Software**
  - Modelo Cascata
  - Modelo RAD
  - Modelos Evolutivos
  - Métodos Ágeis

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.
- É escolhido com base:
  - Na natureza do projeto e da aplicação.
  - Nos métodos e ferramentas a serem utilizados.
  - Nos controles e produtos que precisam ser entregues.

Modelos de Processo de Software

- Todo desenvolvimento de software pode ser caracterizado como um laço de solução:



Modelos de Processo de Software

- Alguns exemplos:
  - Modelo Cascata
  - Modelo RAD
  - Modelos Evolutivos (ou Evolucionários)
    - Modelo Incremental
    - Prototipação
    - Espiral
  - Métodos Ágeis

Roteiro

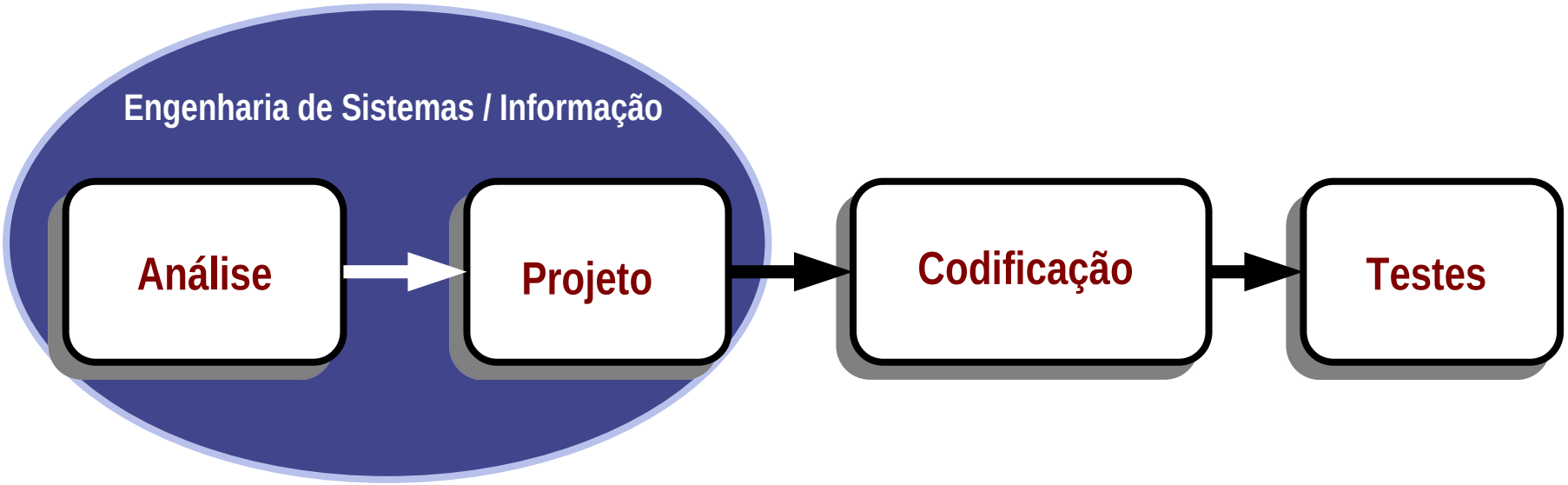
- Fases Genéricas de um Processo de Software.
- Modelos de Processo de Software
  - **Modelo Cascata**
  - Modelo RAD
  - Modelos Evolutivos
  - Métodos Ágeis

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, **sequencial** ao desenvolvimento de software.

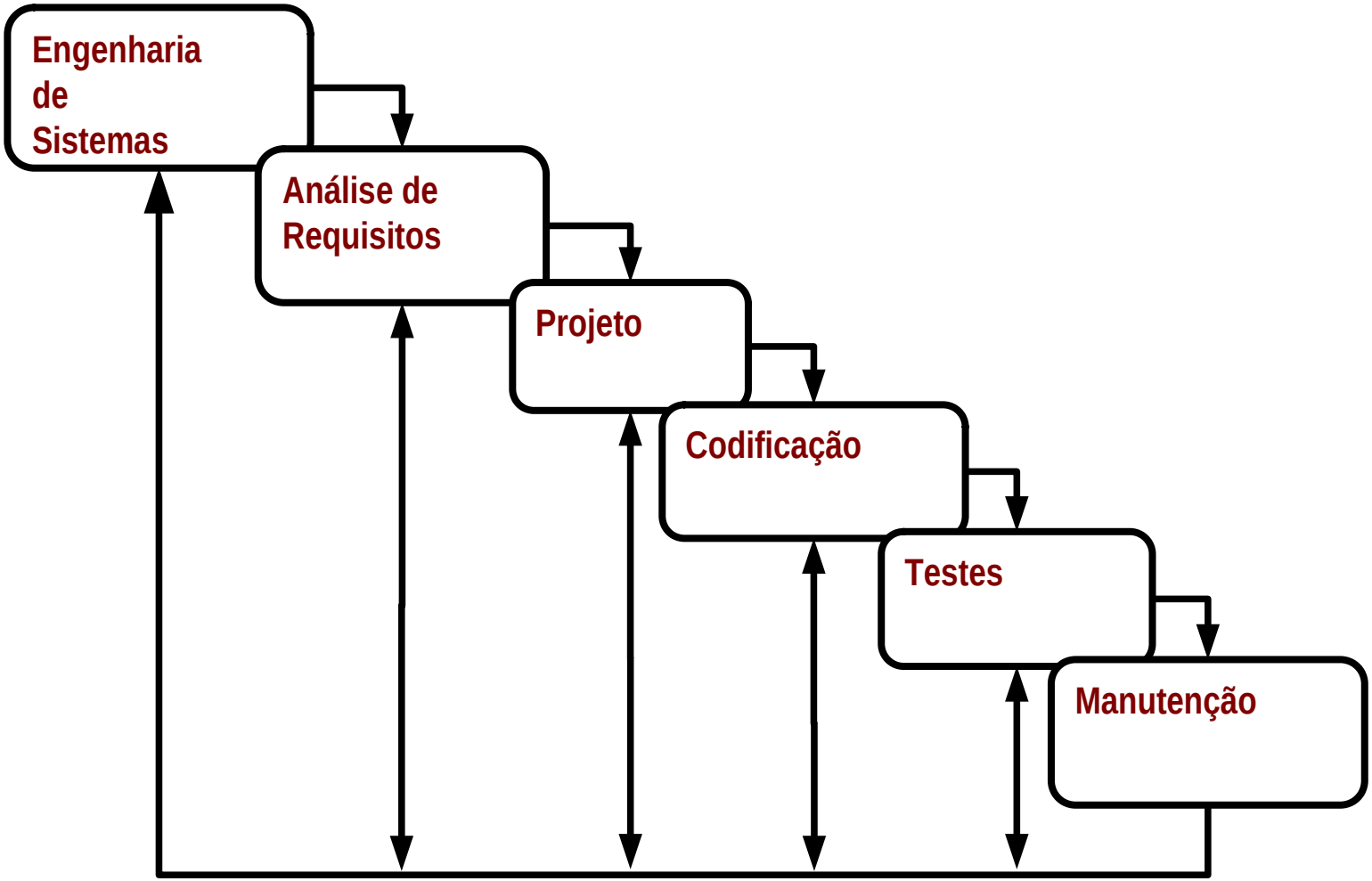
Modelo Cascata

- Muitas organizações que usam esse modelo, aplicam-no de forma estritamente linear.

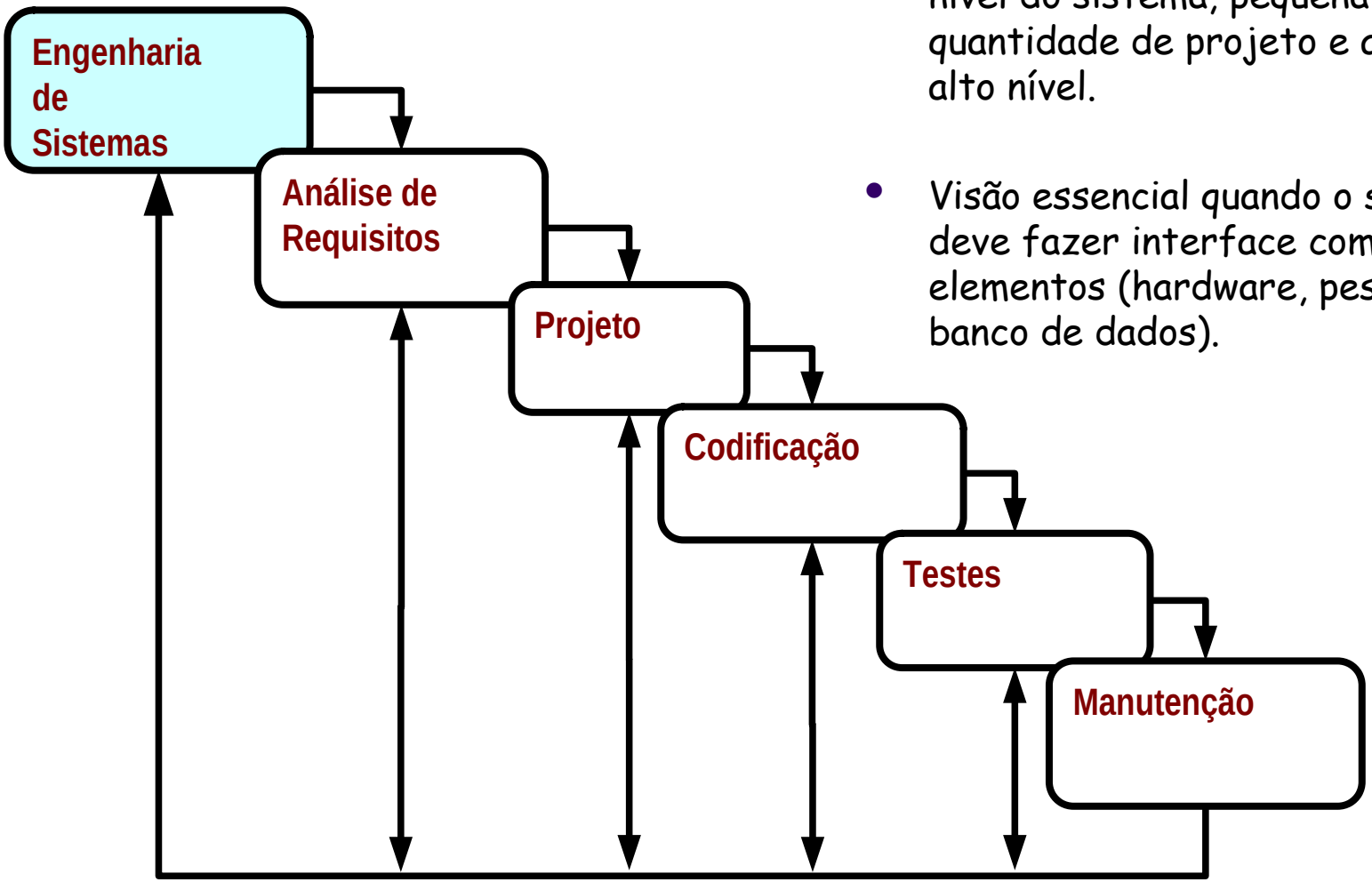


Modelo Cascata

- O processo pode prever *feedback*.

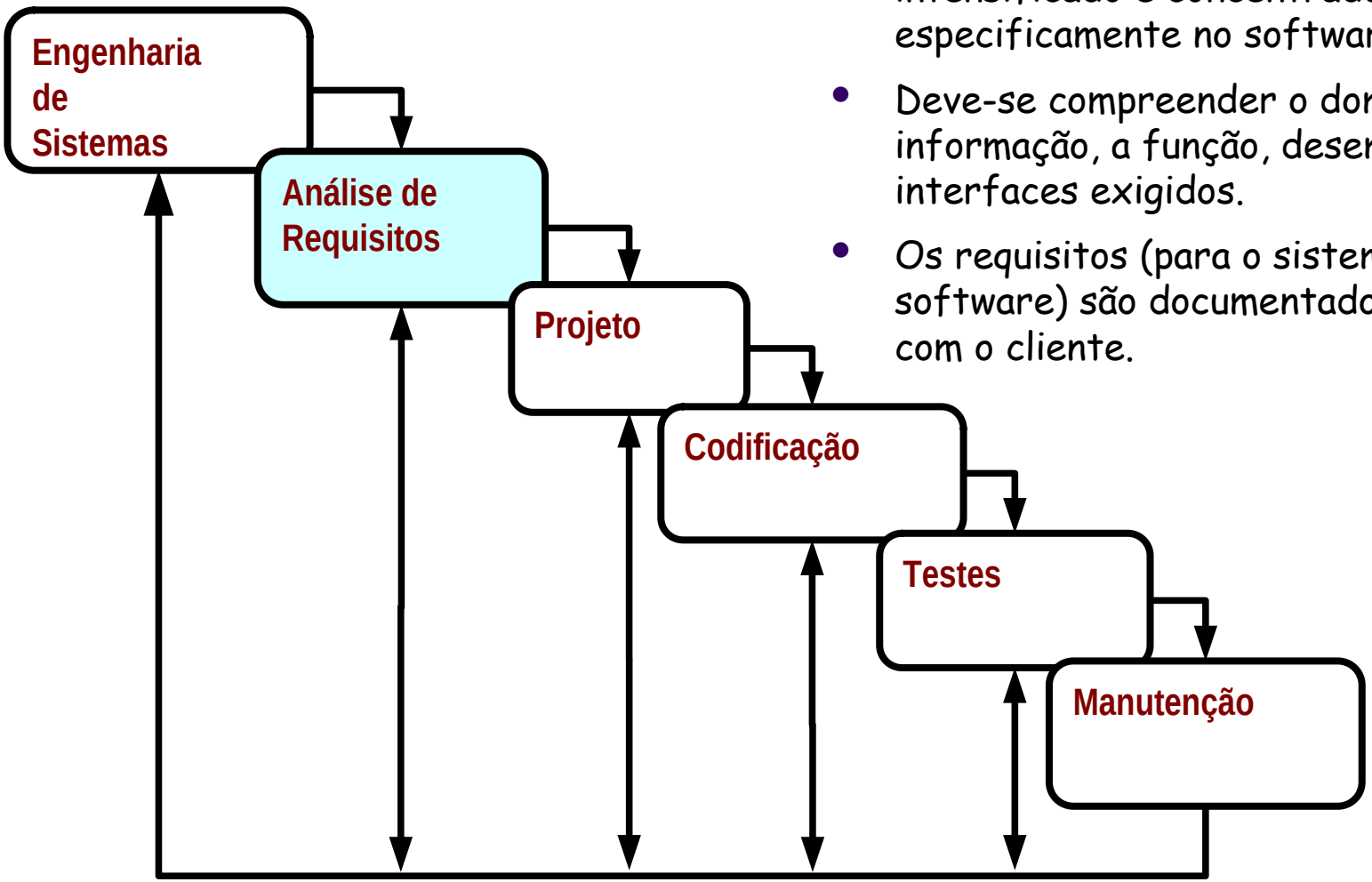


Atividades do Modelo Cascata



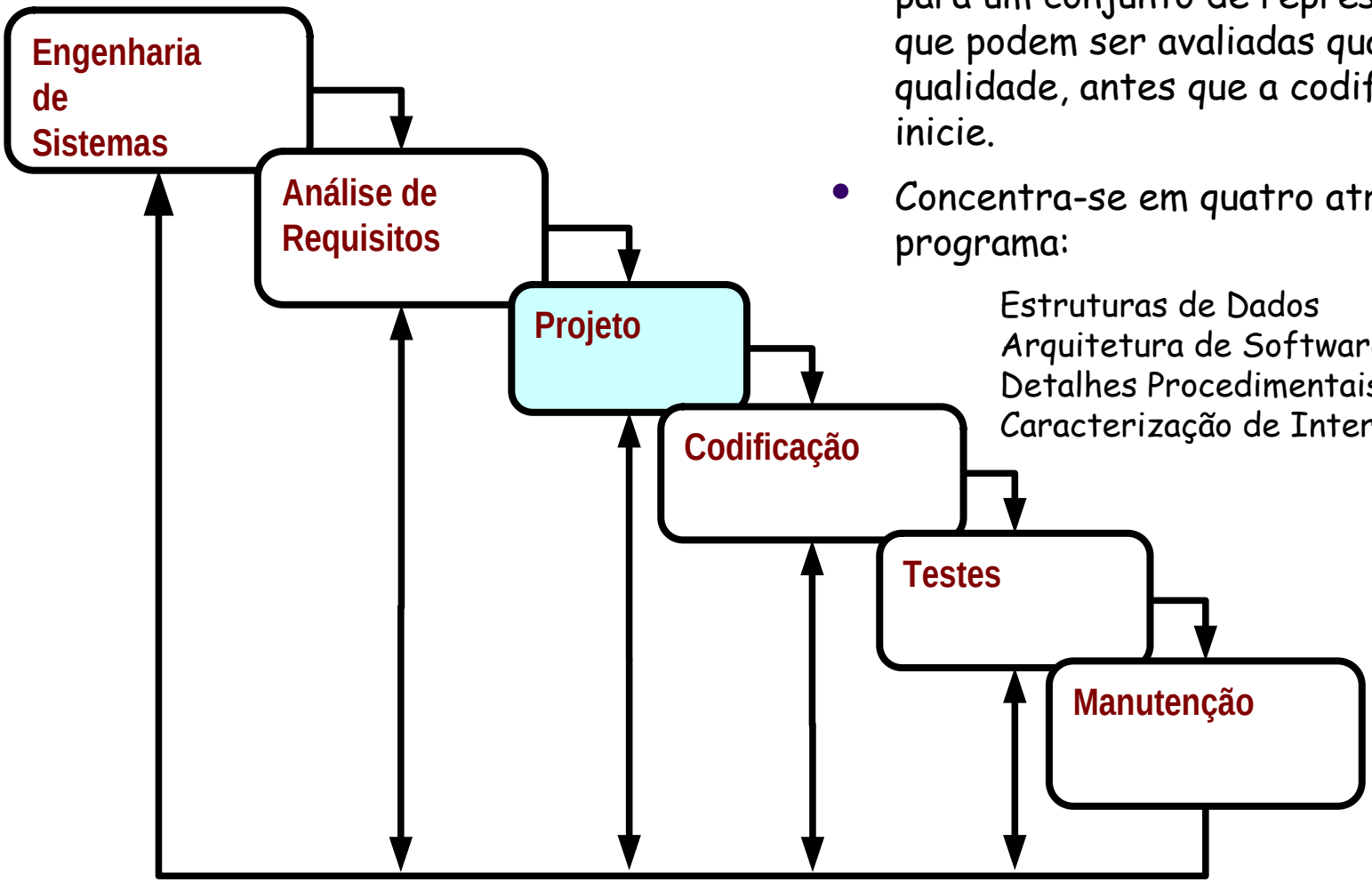
- Envolve a coleta de requisitos em nível do sistema, pequena quantidade de projeto e análise de alto nível.
- Visão essencial quando o software deve fazer interface com outros elementos (hardware, pessoas e banco de dados).

Atividades do Modelo Cascata



- 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.

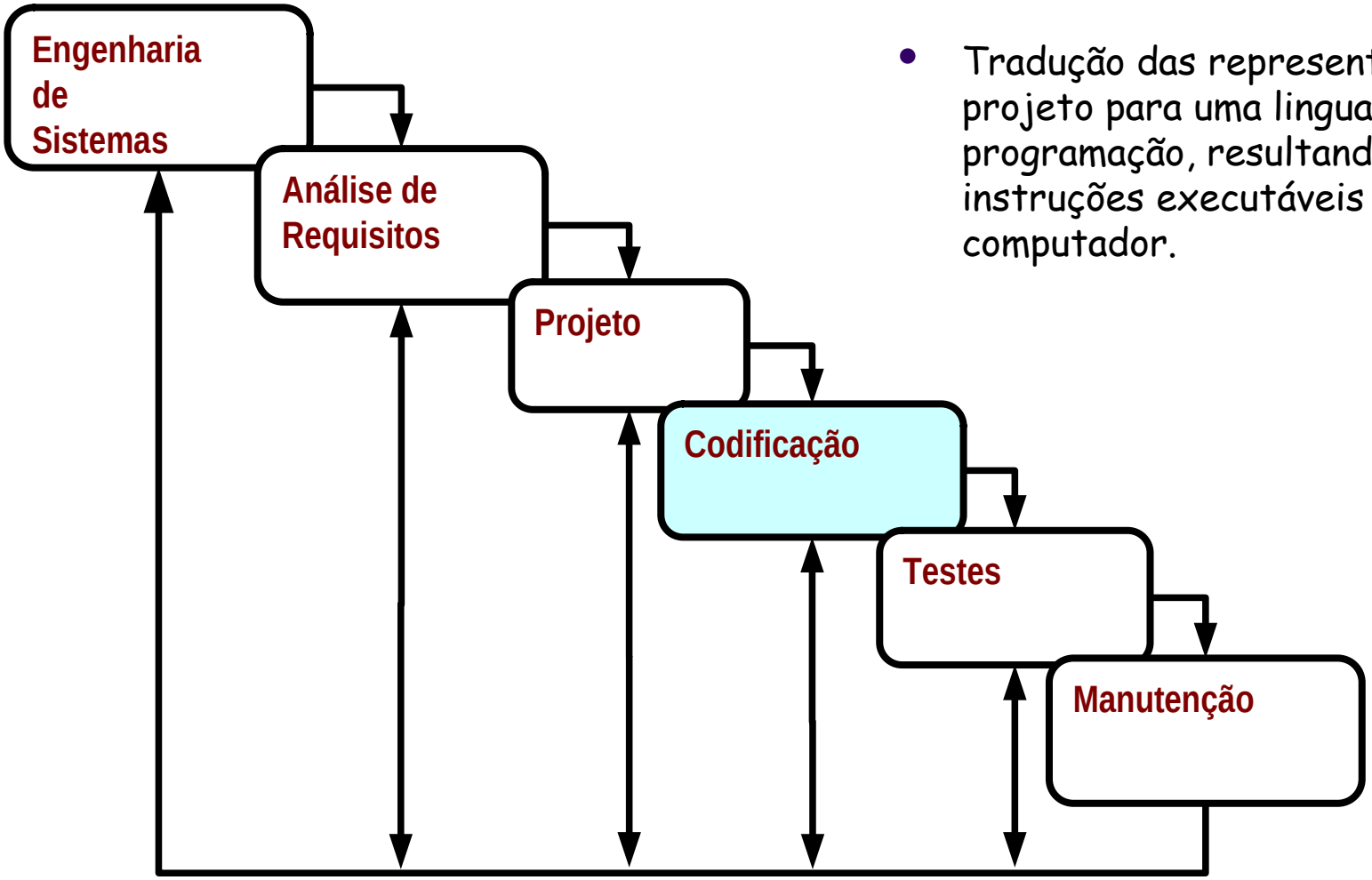
Atividades do Modelo Cascata



- Tradução dos requisitos do software para um conjunto de representações que podem ser avaliadas quanto à qualidade, antes que a codificação se inicie.
- Concentra-se em quatro atributos do programa:

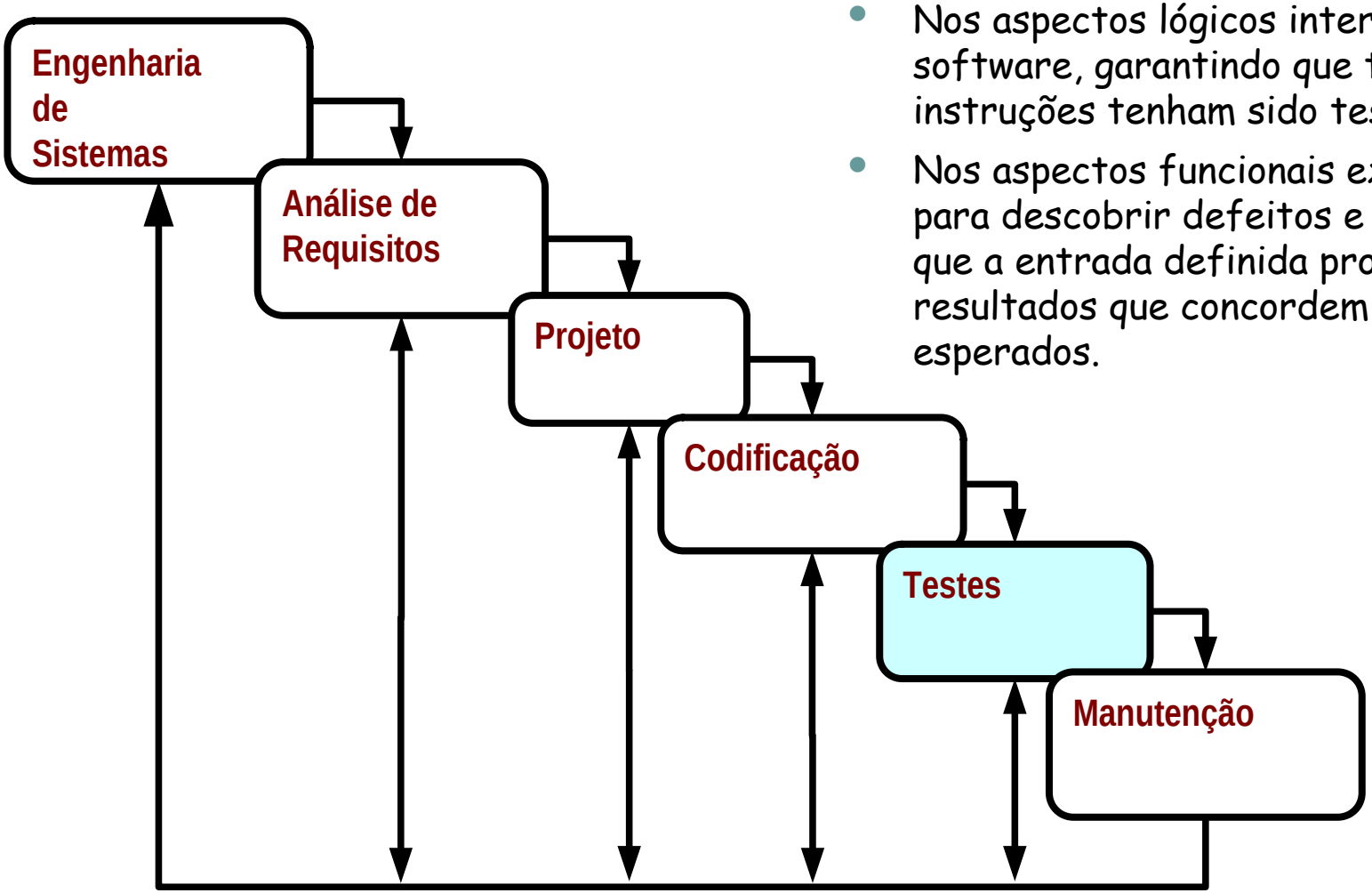
  Estruturas de Dados
  Arquitetura de Software
  Detalhes Procedimentais
  Caracterização de Interfaces

Atividades do Modelo Cascata



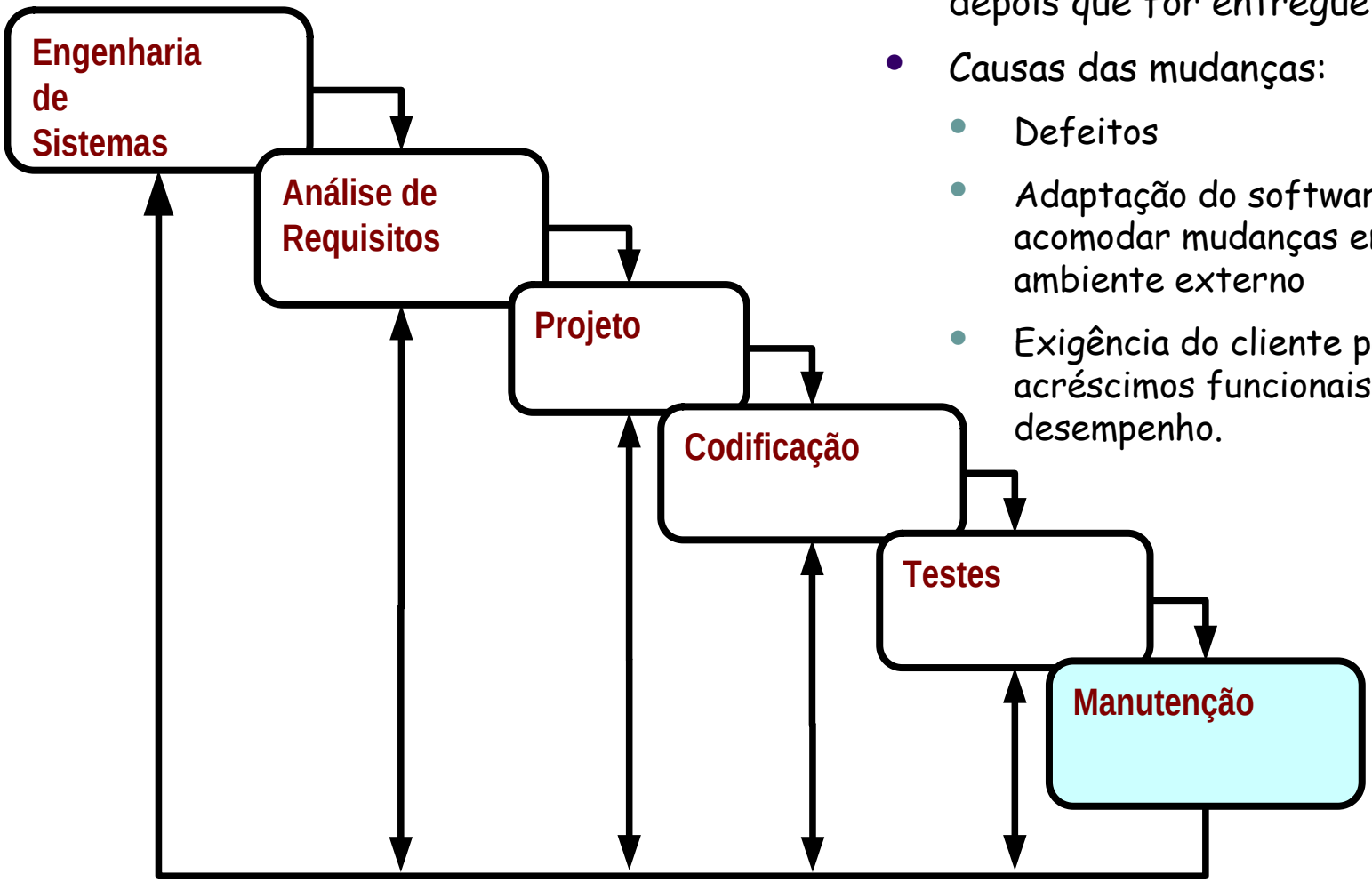
- Tradução das representações do projeto para uma linguagem de programação, resultando em instruções executáveis pelo computador.

Atividades do Modelo Cascata



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

Atividades do Modelo Cascata



- O software deverá sofrer mudanças depois que for entregue ao cliente.
- Causas das mudanças:
  - Defeitos
  - Adaptação do software para acomodar mudanças em seu ambiente externo
  - Exigência do cliente para acréscimos funcionais e de desempenho.

Problemas com o Modelo Cascata

- Projetos reais raramente seguem o fluxo sequencial que o modelo propõe.
- É difícil estabelecer explicitamente no início 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.
- muitas vezes os desenvolvedores ficam ociosos desnecessariamente, devido a estados de bloqueio (dependência entre tarefas).

Problemas com o Modelo Cascata

- Projetos reais raramente seguem o fluxo sequencial que o modelo propõe.
- É difíc… …os requis… …ma incert…
- O clie… …do softw… desen…
- muitas… desnecessariamente, devido a estados de bloqueio (dependência entre tarefas).

Embora o Modelo Cascata (ou Sequencial) tenha fragilidades, ele é significativamente melhor do que uma abordagem casual (caótica) ao desenvolvimento de software.

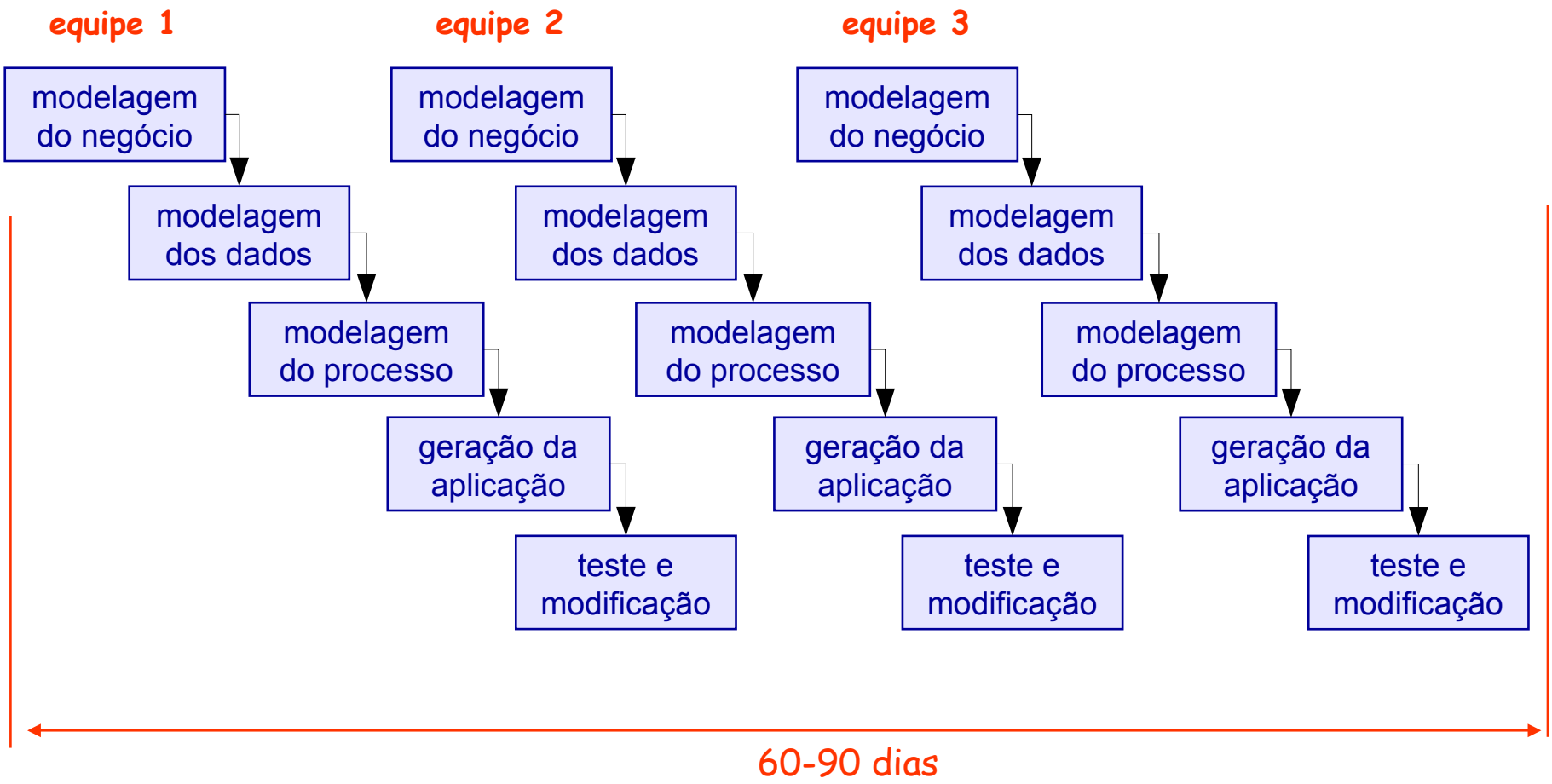
Roteiro

- Fases Genéricas de um Processo de Software.
- Modelos de Processo de Software
  - Modelo Cascata
  - **Modelo RAD**
  - Modelos Evolutivos
  - Métodos Ágeis

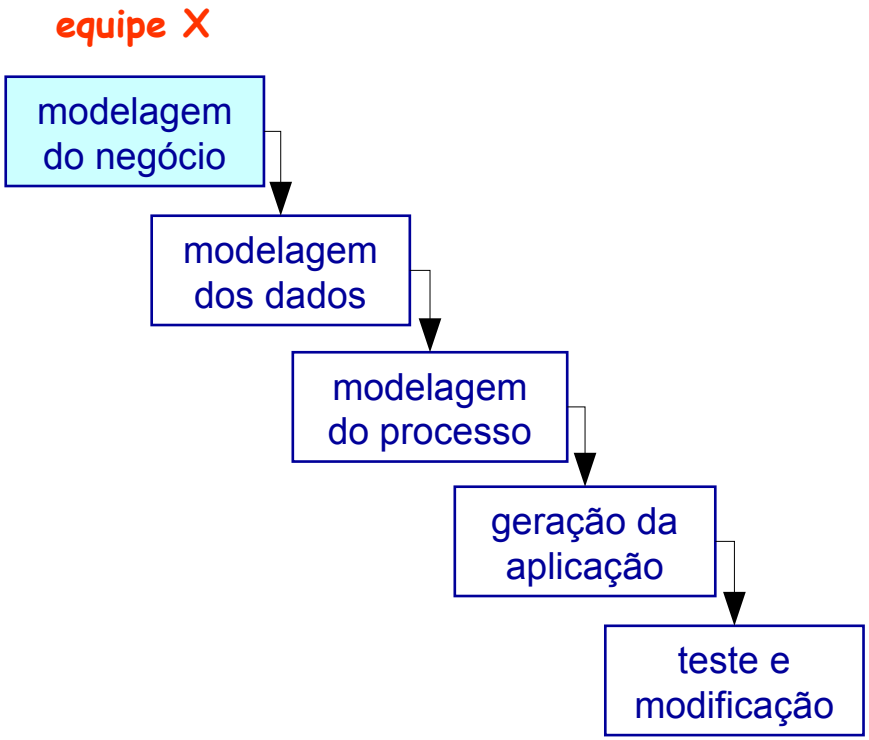
Leitura Complementar:
Modelo RAD
(*Rapid Application Development*)

- Proposto por James Martin na IBM.
- Desenvolvimento rápido de aplicações.
- É o modelo sequencial linear, porém que enfatiza um desenvolvimento extremamente rápido.
- A "alta velocidade" é conseguida através de uma abordagem de construção baseada em componentes.
- Usado quando os requisitos são bem definidos e o escopo do sistema é restrito.

Leitura Complementar:
Modelo RAD
(*Rapid Application Development*)

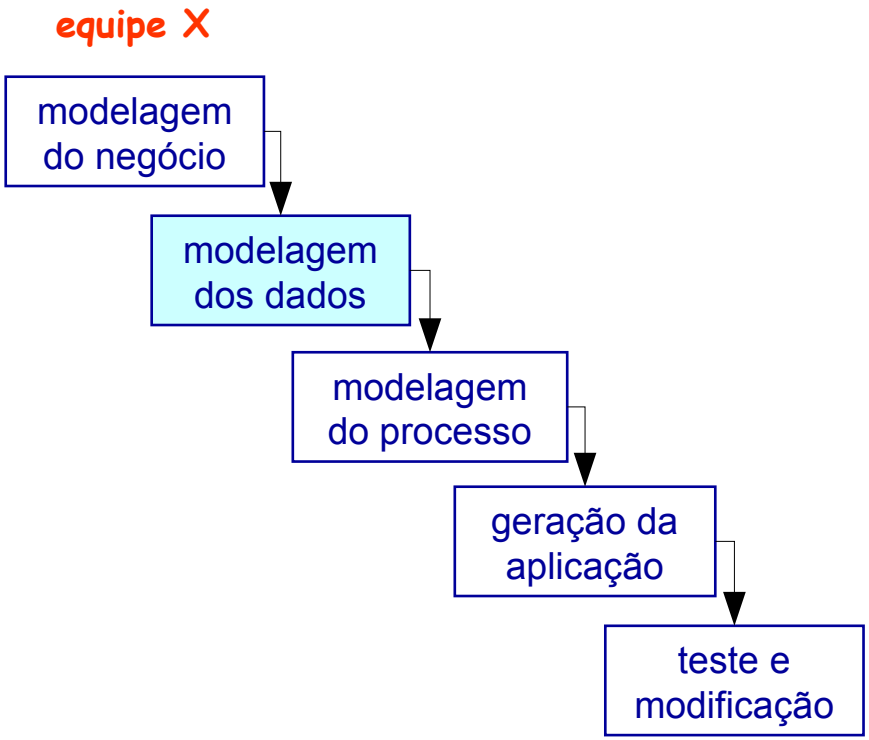


Leitura Complementar:
Modelo RAD
(*Rapid Application Development*)



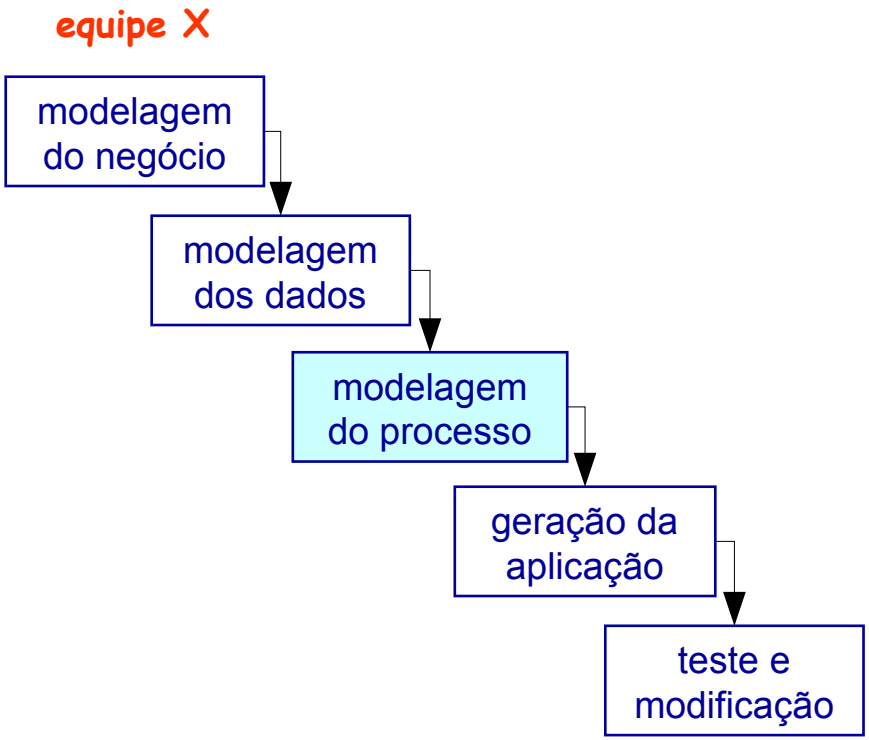
- O fluxo de informação entre as funções do negócio é modelado de maneira a responder as seguintes questões:
  - que informação dirige o processo do negócio?
  - que informação é gerada?
  - quem gera a informação?
  - para onde a informação vai?
  - quem a processa?

Leitura Complementar:
Modelo RAD
(*Rapid Application Development*)



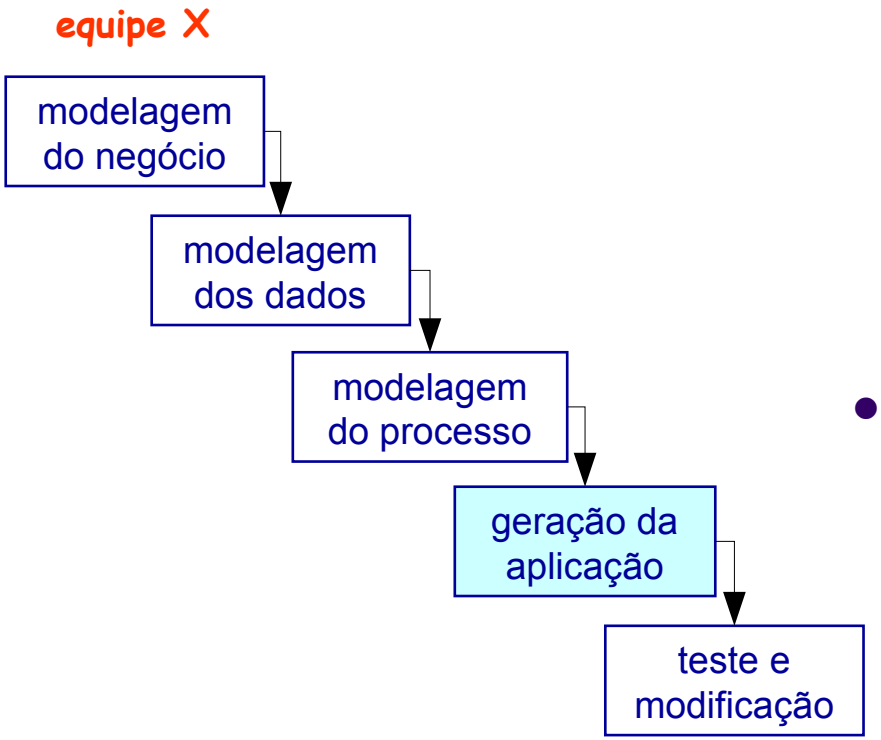
- O fluxo de informação definido na fase anterior é refinado em um conjunto de objetos de dados que são necessários para dar suporte ao negócio.
- São identificadas as características de cada objeto e são definidos seus relacionamentos.

Leitura Complementar:
Modelo RAD
(*Rapid Application Development*)



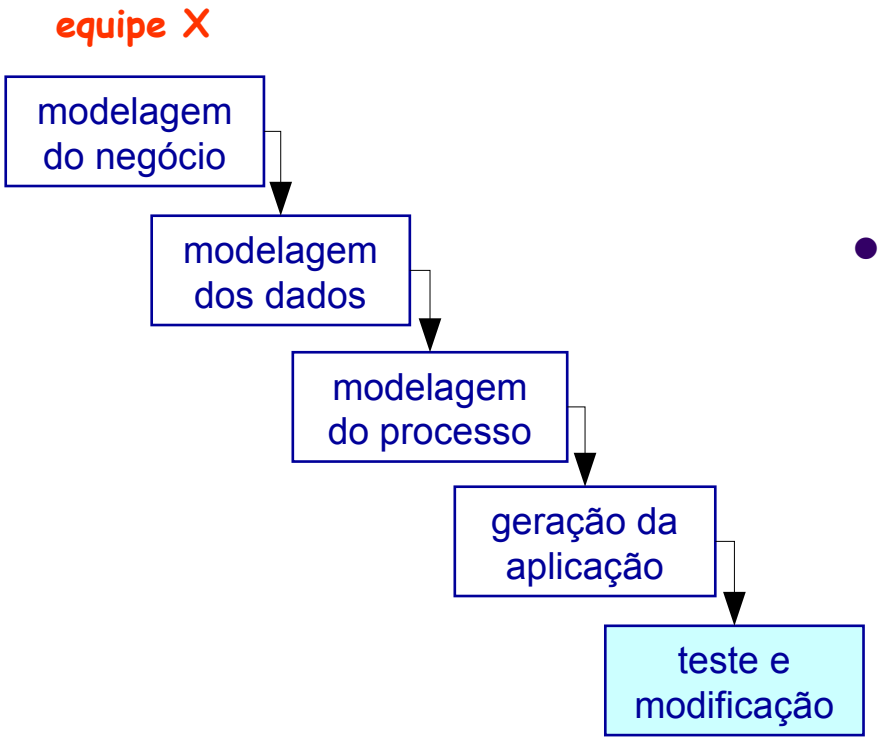
- Os objetos de dados definidos são transformados para se obter o fluxo de informação necessário para implementar uma função do negócio.
- São criadas as descrições dos processamentos necessários para manipular esses objetos de dados.

Leitura Complementar:
Modelo RAD
(*Rapid Application Development*)



- O modelo RAD:
  - assume o uso de técnicas de 4a. geração; e
  - reusa componentes quando possível ou cria componentes reutilizáveis.
- Ferramentas automatizadas são utilizadas para gerar software.

Leitura Complementar:
Modelo RAD
(*Rapid Application Development*)



- Por reutilizar componentes, muitas vezes eles já foram testados, o que reduz o tempo de teste.
- Os novos componentes devem ser testados e as interfaces devem ser exercitadas.

Leitura Complementar:
Quando Usar o Modelo RAD?

- As restrições de tempo impostas pelo projeto demandam um escopo de escala.
- A aplicação pode ser modularizada de forma que cada grande função possa ser completada em menos de 3 meses.
- Cada grande função pode ser alocada para uma equipe distinta para depois serem integradas para formar o todo.

Leitura Complementar:
Problemas com o Modelo RAD

- Para projetos escaláveis, mas grandes, o RAD requer recursos humanos suficientes para criar um número adequado de equipes.
- RAD requer um comprometimento entre desenvolvedores e clientes para que as atividades possam ser realizadas rapidamente e o sistema seja concluído em um tempo abreviado.
- Se o comprometimento for abandonado por qualquer das partes, o projeto falhará.
- Não é apropriado quando os riscos são grandes.

Roteiro

- Fases Genéricas de um Processo de Software.
- Modelos de Processo de Software
  - Modelo Cascata
  - Modelo RAD
  - **Modelos Evolutivos**
  - Métodos Ágeis

Modelos de Processo Evolutivos

- Modelos que acomodam o aspecto evolutivo do software:
  - Os requisitos do produto e do negócio frequentemente mudam durante o desenvolvimento do software.
  - O uso de uma linha sequencial de desenvolvimento (modelo cascata) não é realista.
- Modelos evolutivos são iterativos: permitem que o engenheiro desenvolva incrementalmente uma versão mais completa do software.

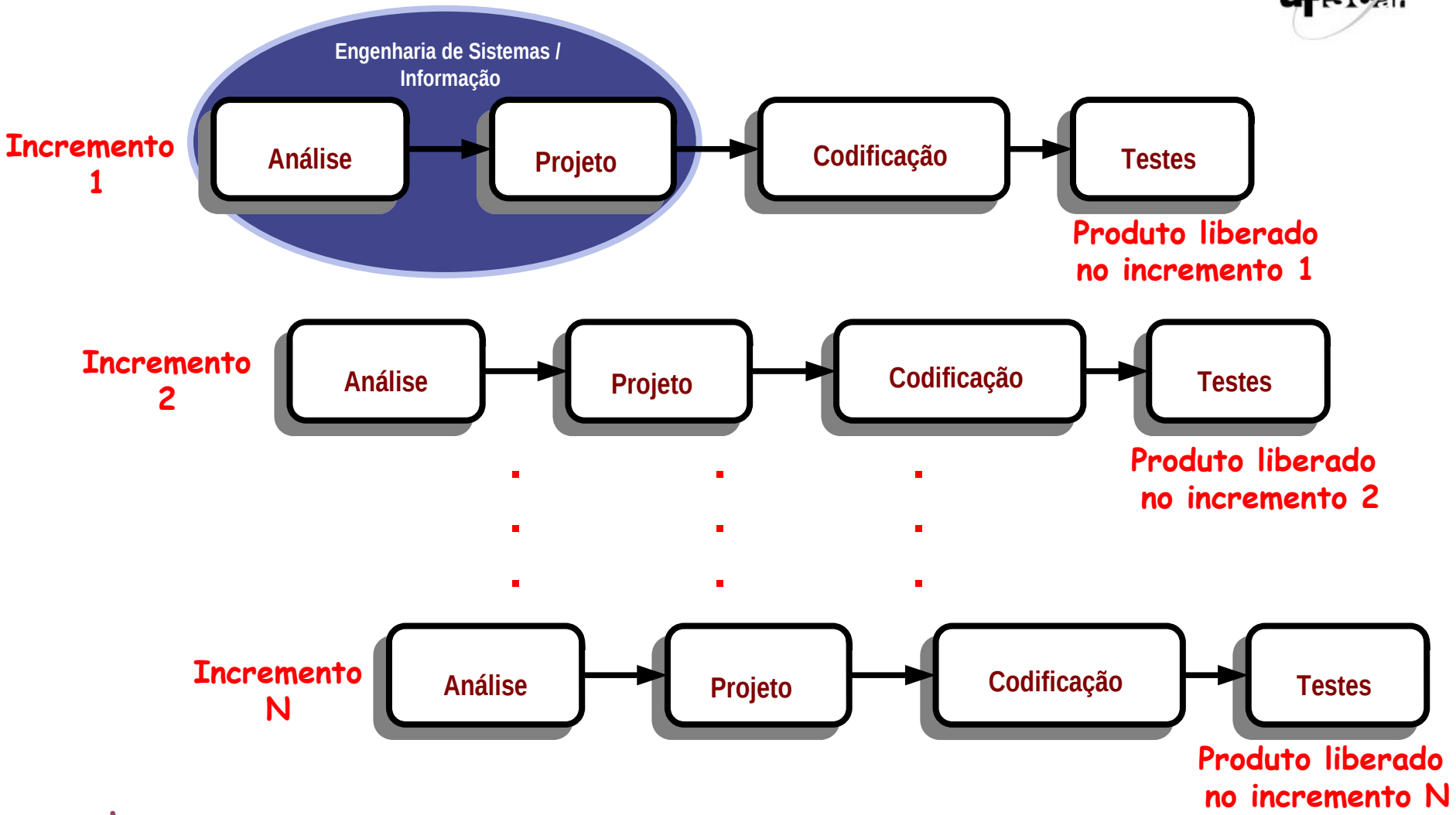
Modelos de Processo Evolutivos

- Alguns modelos evolutivos:
  - Incremental
  - Prototipação
  - Espiral
  - Modelo Baseado em Componentes

Leitura Complementar:
Modelo Incremental

- Permite que um **conjunto limitado** de funcionalidades seja fornecida ao usuário.
  - A cada incremento produz uma **versão operacional** do software.
- A cada passo, as funcionalidades são refinadas e novas são disponibilizadas.
  - O processo **se repete** até que um produto completo seja produzido.

Leitura Complementar:
Modelo Incremental

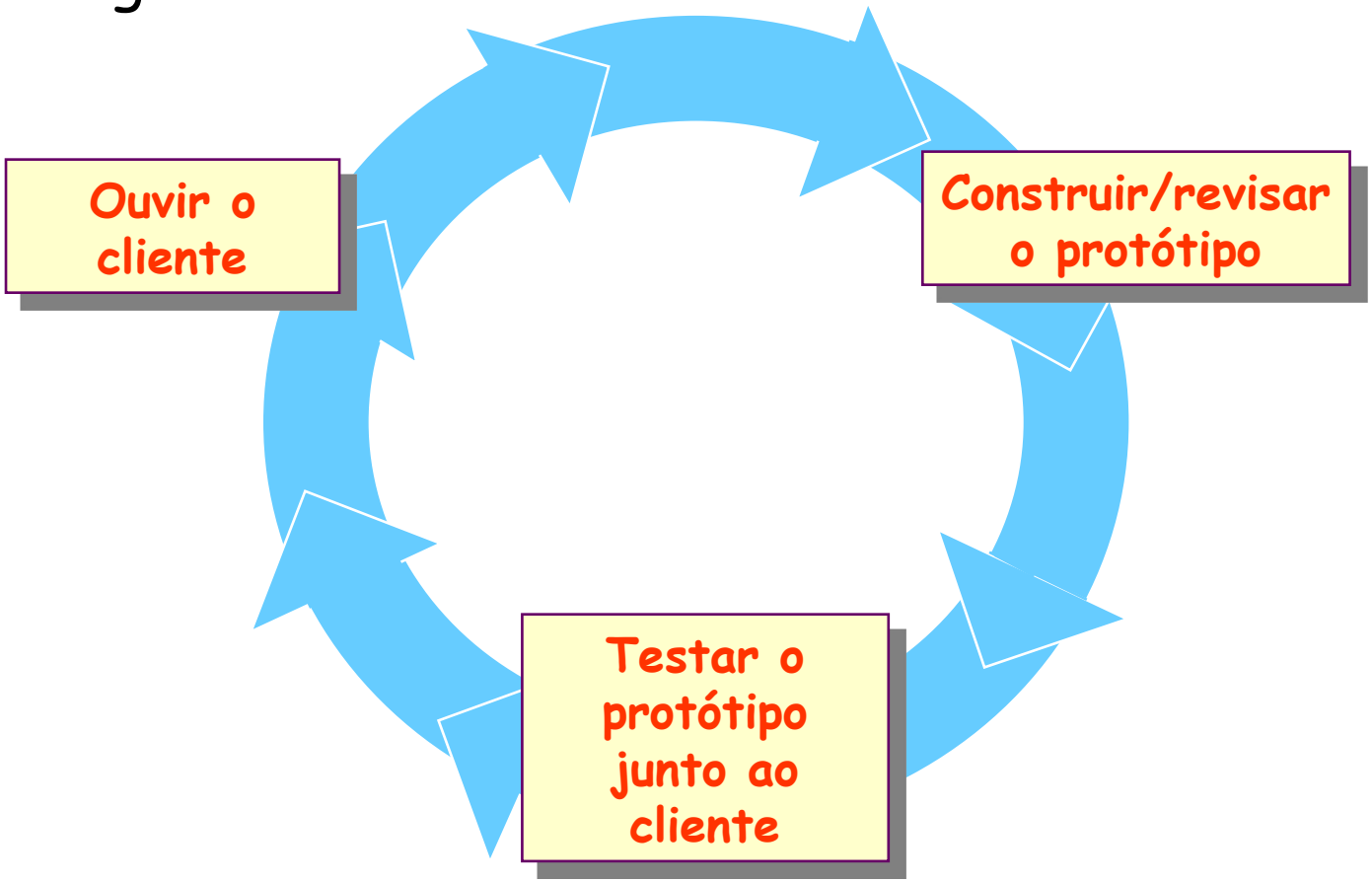


Modelo de Prototipação

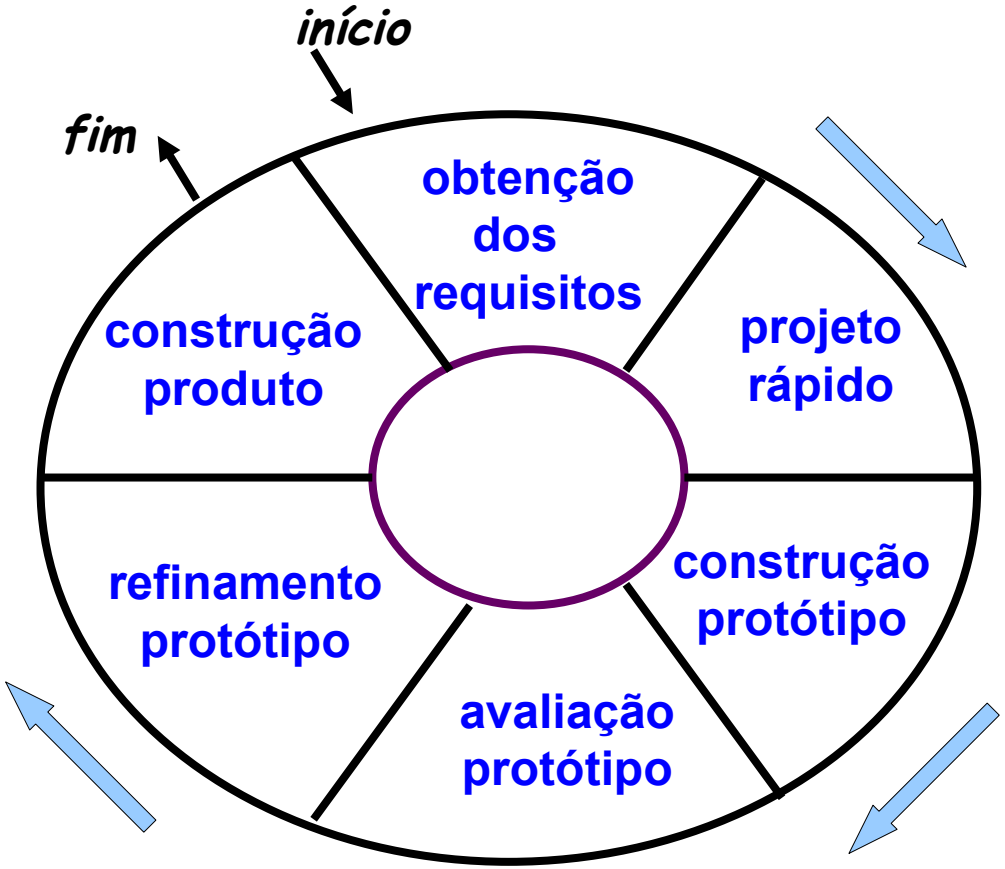
- Modelo de processo que possibilita que o desenvolvedor crie **um modelo do software** que deve ser construído.
- Idealmente, o modelo (**protótipo**) serve como um mecanismo para identificar os requisitos de software.
- É apropriado para quando o cliente definiu um conjunto de objetivos gerais para o software, mas não tem noções dos detalhes.

Modelo de Prototipação

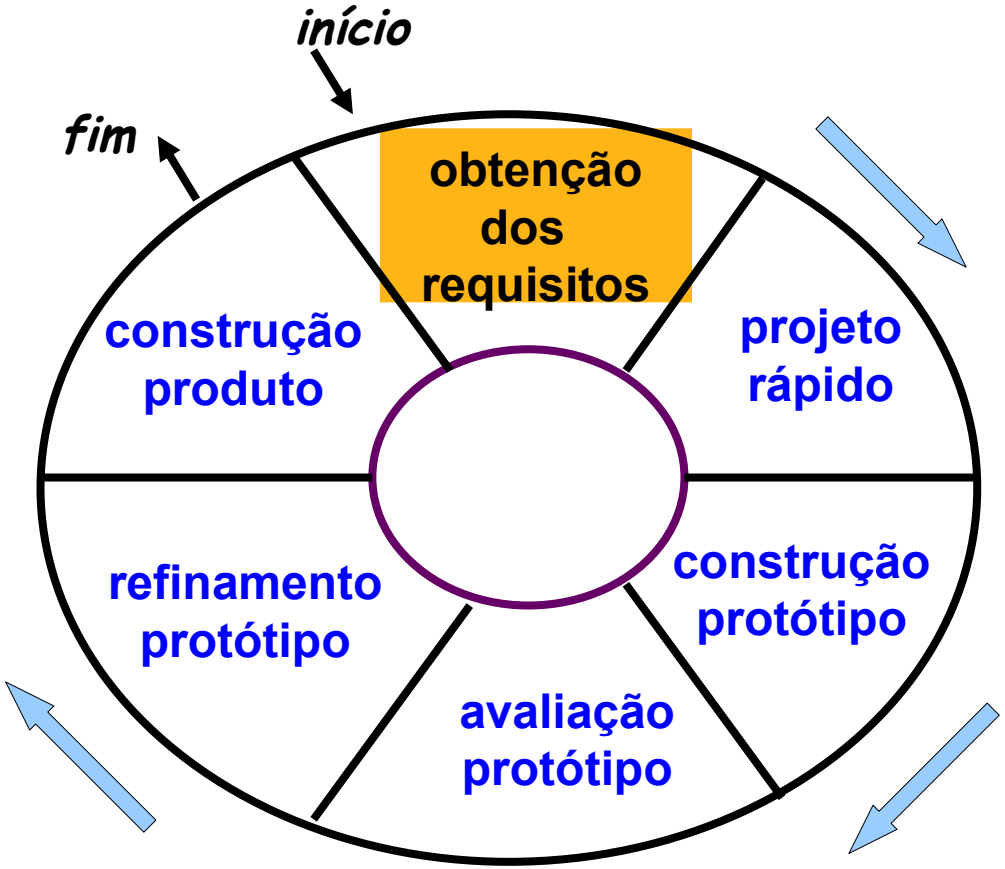
- Idéia geral:



Modelo de Prototipação: Fases

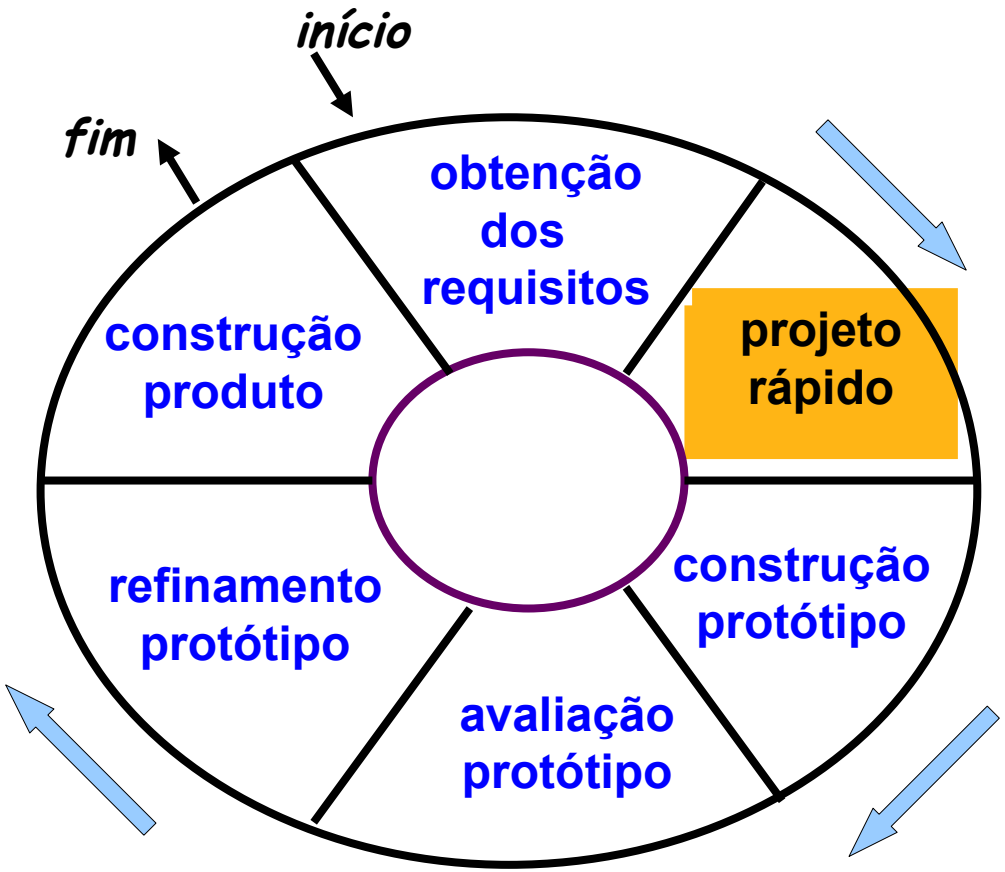


Modelo de Prototipação: Fases



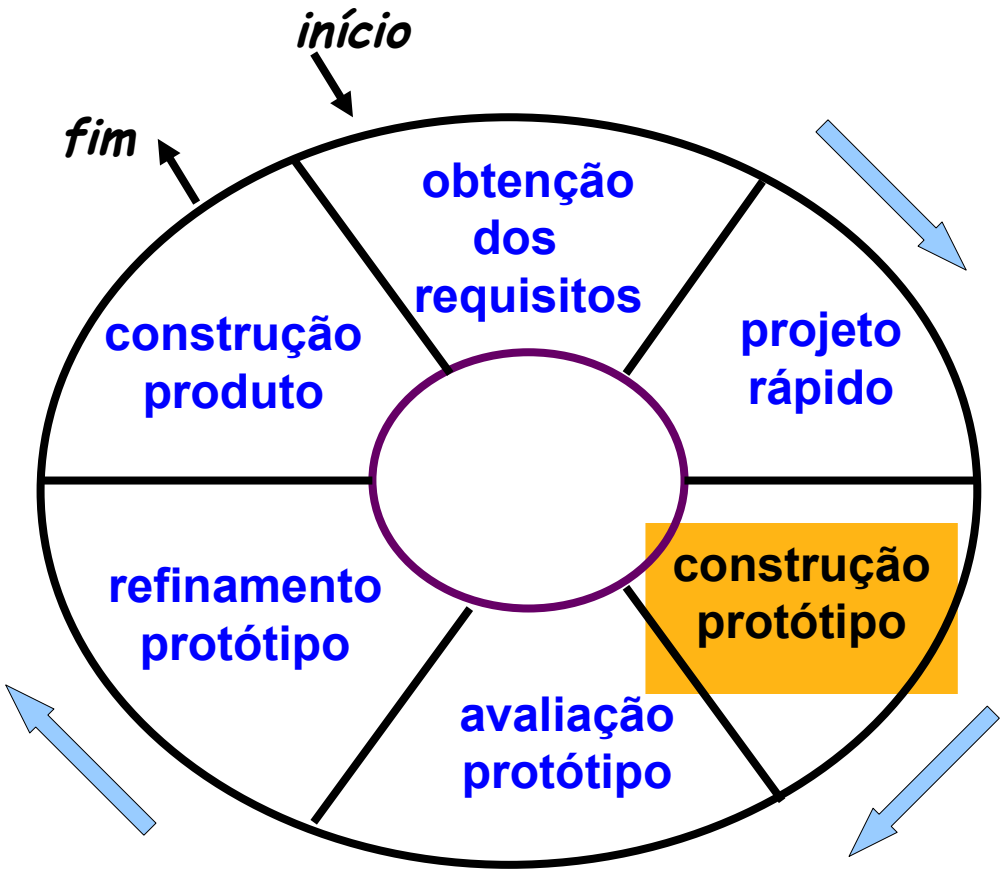
- Desenvolvedor e cliente:
  - Definem os objetivos gerais do software.
  - Identificam quais requisitos são conhecidos e as áreas que necessitam de definições adicionais.

Modelo de Prototipação: Fases



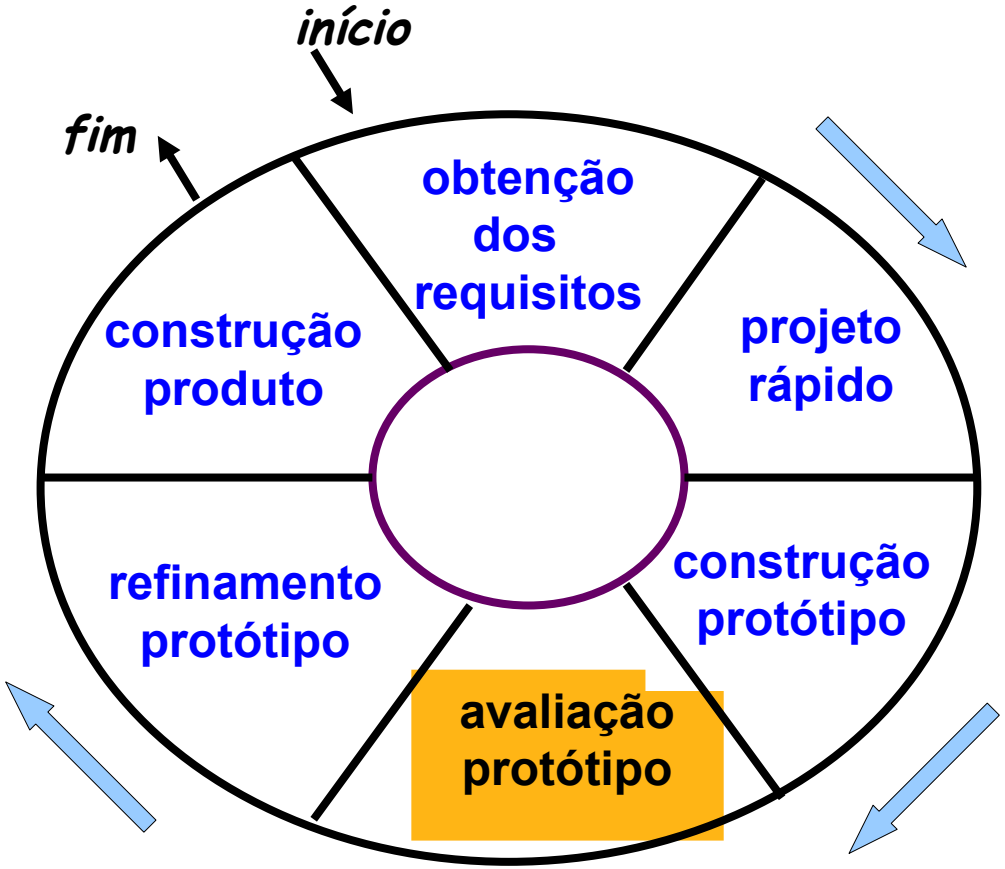
- Representação dos aspectos do software que são visíveis ao usuário (abordagens de entrada e formatos de saída).

Modelo de Prototipação: Fases



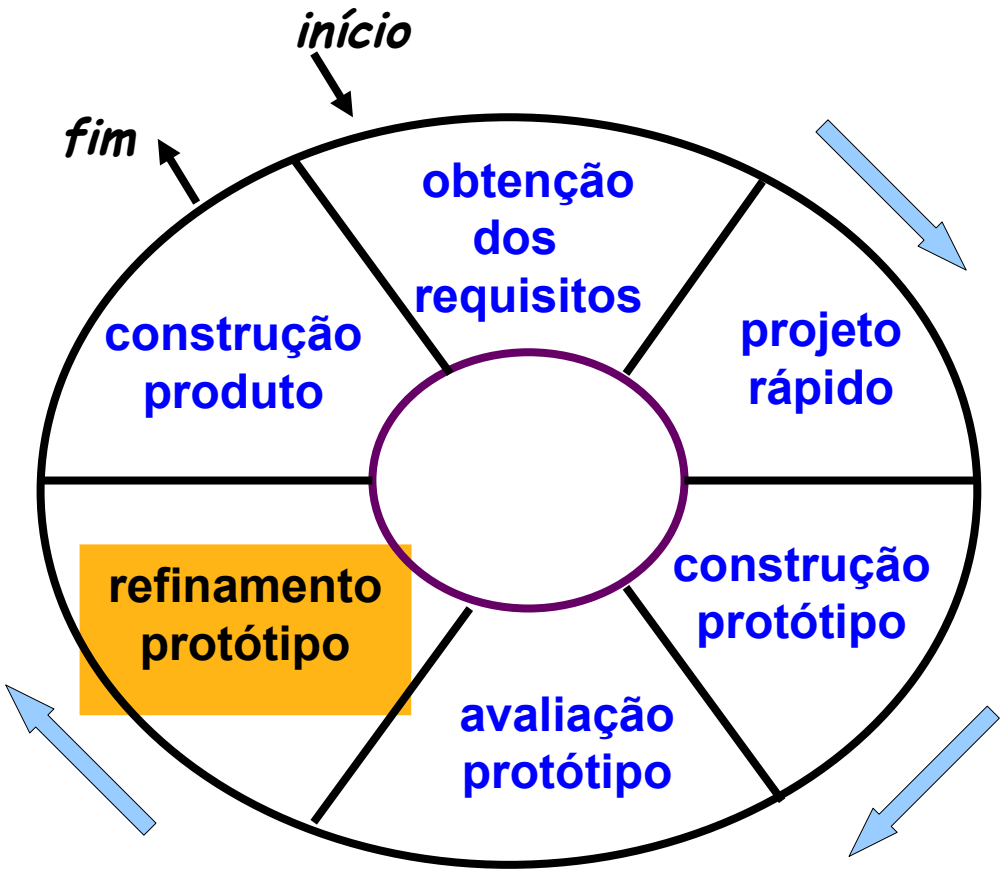
- Implementação rápida do projeto, geralmente baseada em ferramentas de construção de aplicações gráficas.

Modelo de Prototipação: Fases



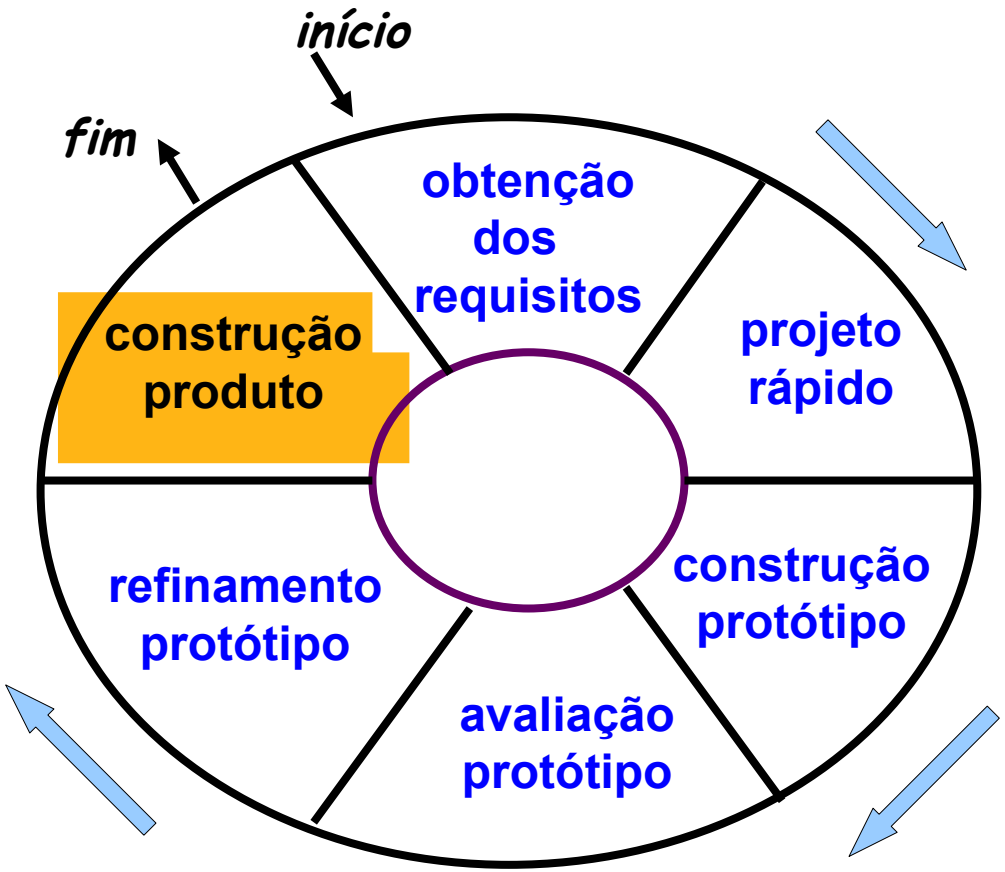
- Cliente e desenvolvedor avaliam o protótipo

Modelo de Prototipação: Fases



- Cliente e desenvolvedor refinam os requisitos do software a ser desenvolvido.
- NOTA: ocorre neste ponto um processo de iteração que pode conduzir à 1a. atividade **até que as necessidades** do cliente sejam **satisfeitas** e o desenvolvedor **compreenda o que precisa ser feito**.

Modelo de Prototipação: Fases



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

Problemas com Prototipação

- O cliente acredita que o protótipo é o produto final.
  - Ele não entende que alguns critérios de qualidade não foram considerados.
- O desenvolvedor frequentemente faz uma implementação comprometida (utilizando o que está disponível) com o objetivo de produzir rapidamente um protótipo.
  - Depois de um tempo ele se familiariza com essas escolhas, e esquece que elas não são apropriadas para o produto final.

Problemas com Prototipação

- O cliente acredita que o protótipo é o produto fi
- de não
- está
- nente
- s
- as para

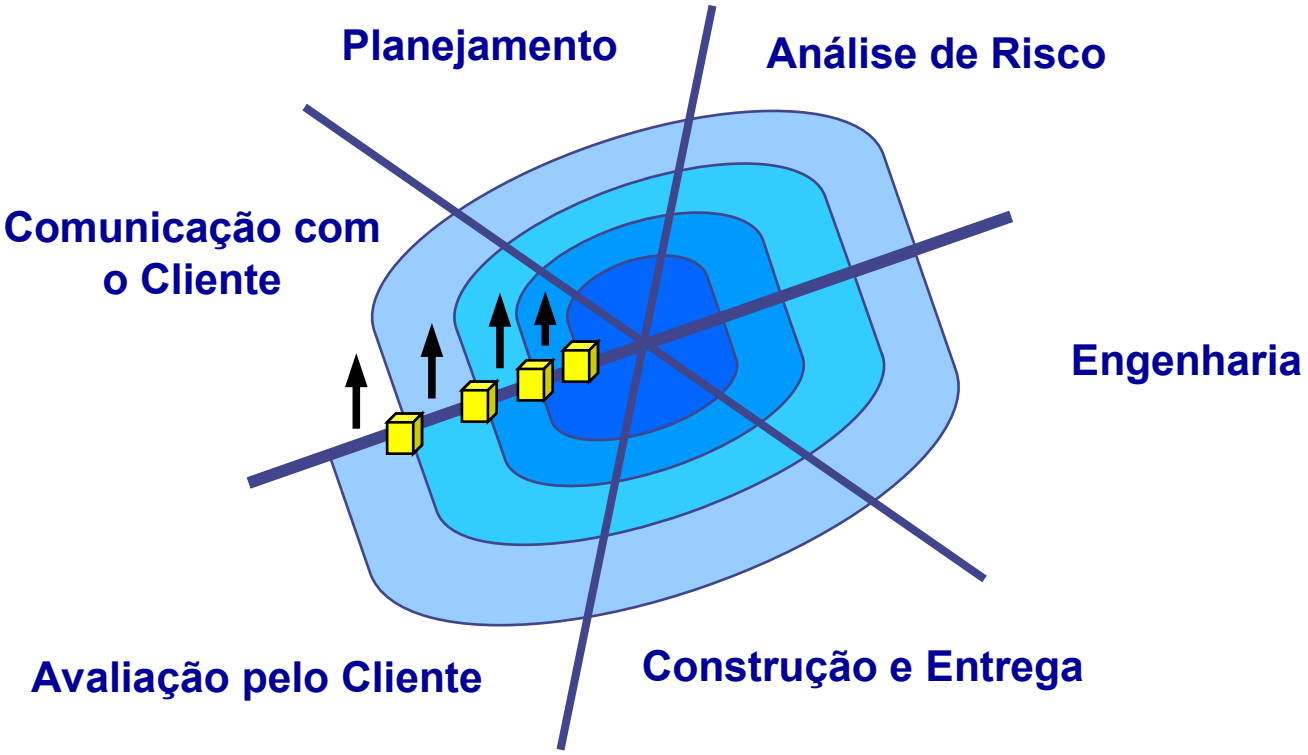
o produto final.

Ainda que possam ocorrer problemas, a prototipação é um ciclo de vida eficiente. A chave é definir as "regras do jogo" logo no começo. O cliente e o desenvolvedor devem concordar que o protótipo serve como um mecanismo para definição dos requisitos.

Modelo Espiral

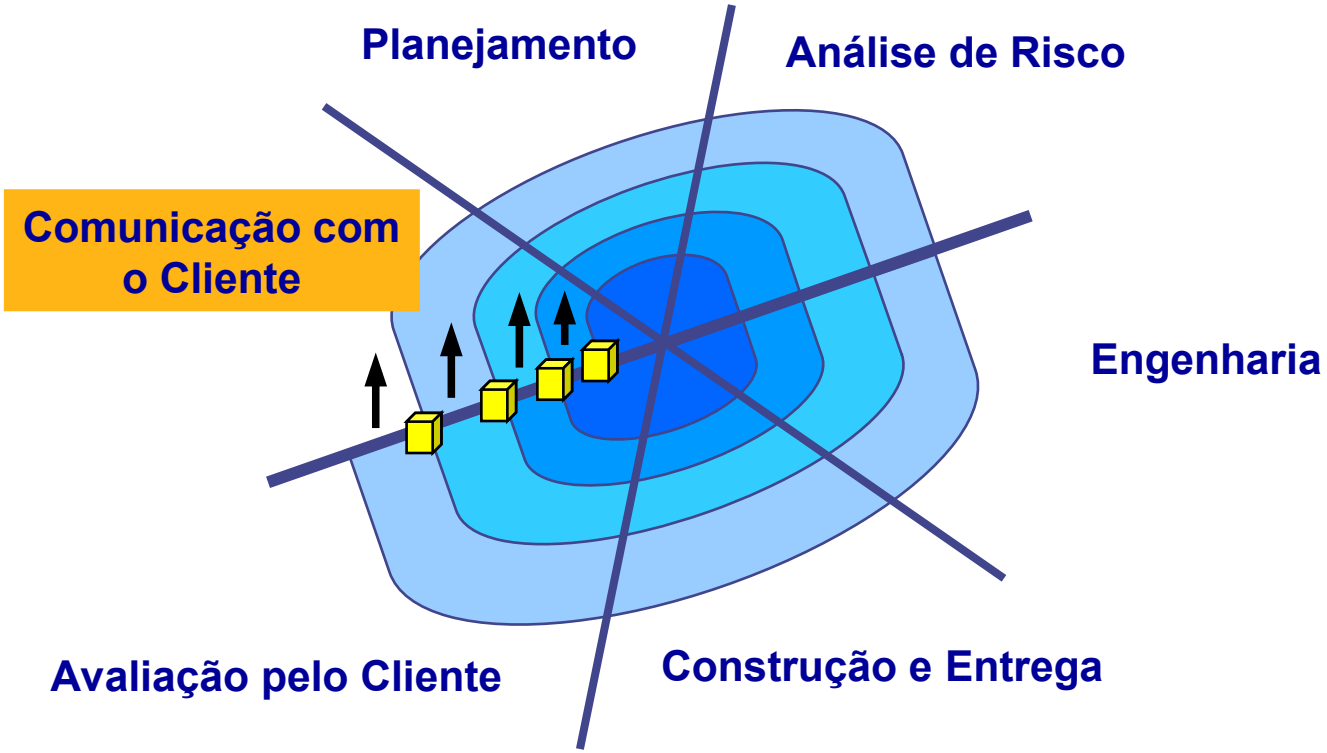
- Originalmente proposto por Barry Boehm.
- Engloba a natureza iterativa da Prototipação com os aspectos sistemáticos e controlados do Modelo Cascata.
- Fornece o potencial para o desenvolvimento rápido de versões incrementais do software.
- Nas primeiras iterações, a versão incremental pode ser um modelo em papel ou um protótipo.
- Nas iterações mais adiantadas, são produzidas versões incrementais mais completas e melhoradas.

Modelo Espiral



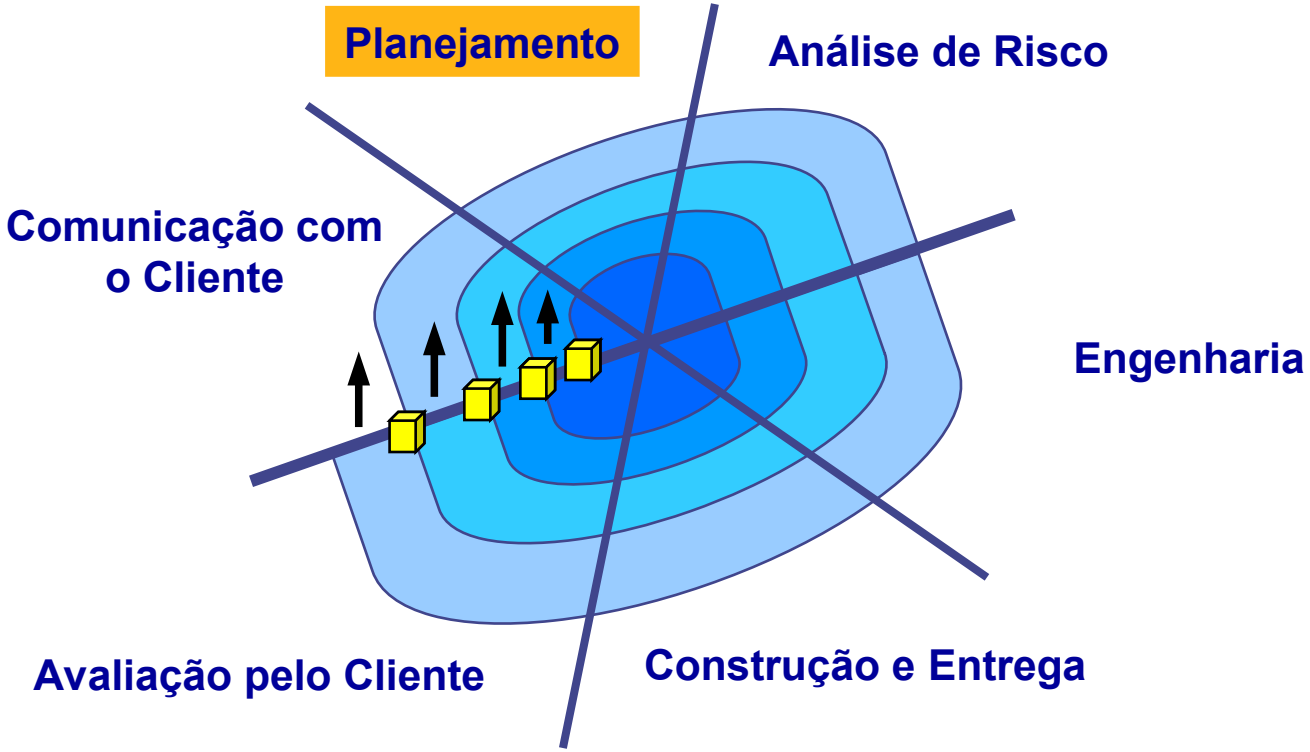
Modelo Espiral

- Tarefas requeridas para estabelecer uma efetiva comunicação entre desenvolvedor e cliente.



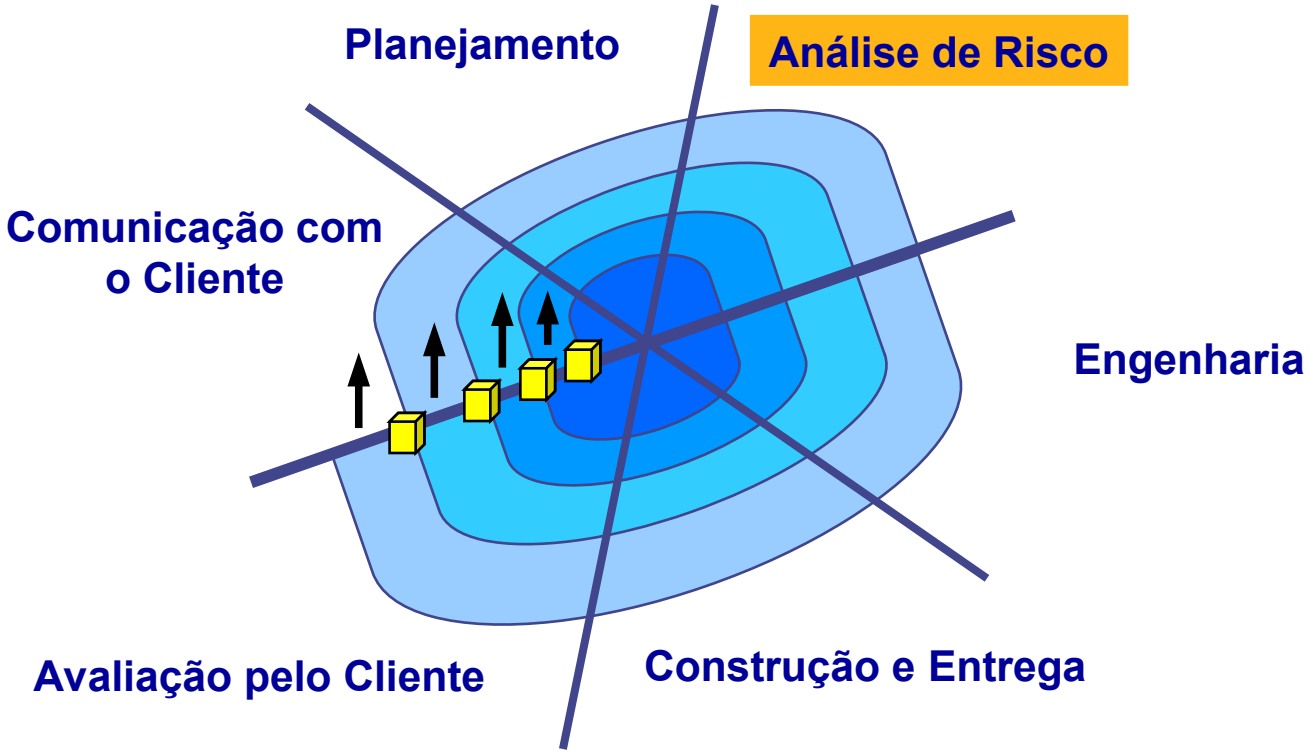
Modelo Espiral

- Tarefas requeridas para definir recursos, referenciais de tempo e outras informações de projeto.



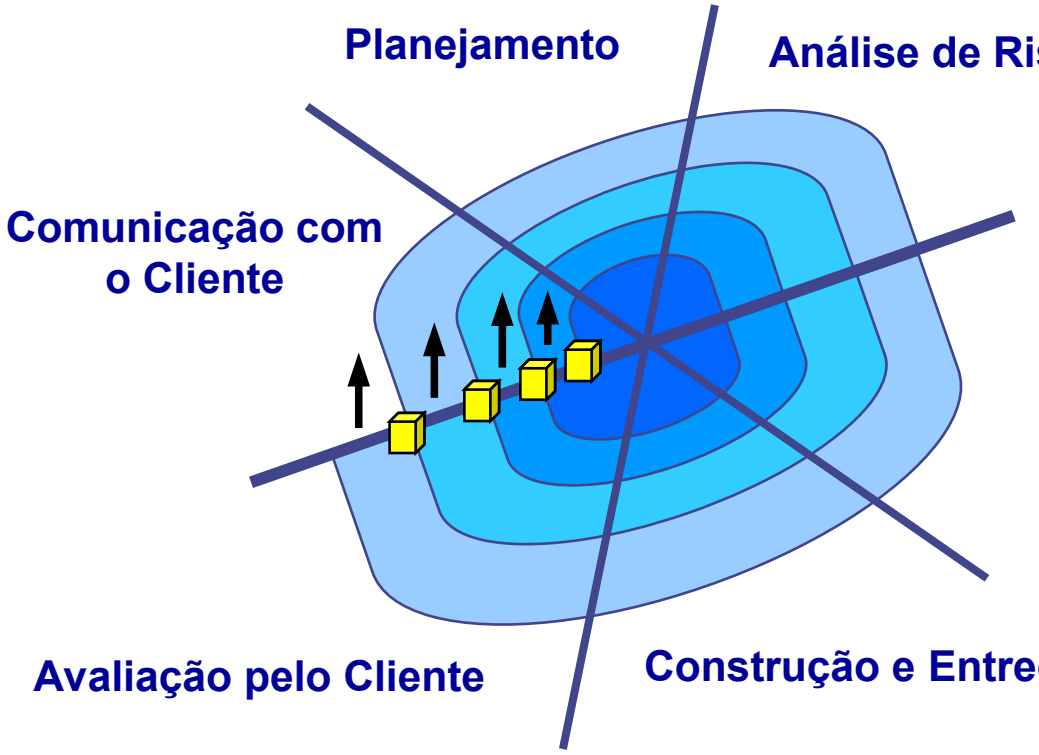
Modelo Espiral

- Tarefas requeridas para fazer levantamento de riscos técnicos e de gerenciamento.



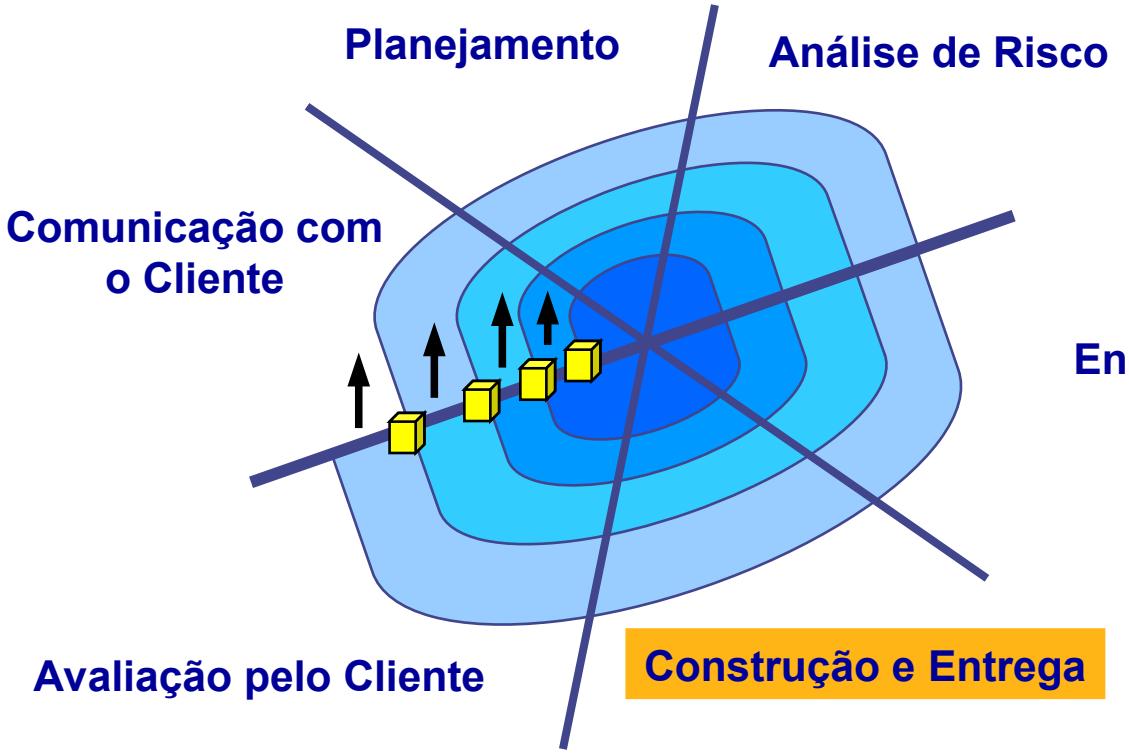
Modelo Espiral

- Tarefas requeridas para construir uma ou mais representações da aplicação.
- Engloba atividades de Engenharia de Sistemas e Engenharia de Requisitos.



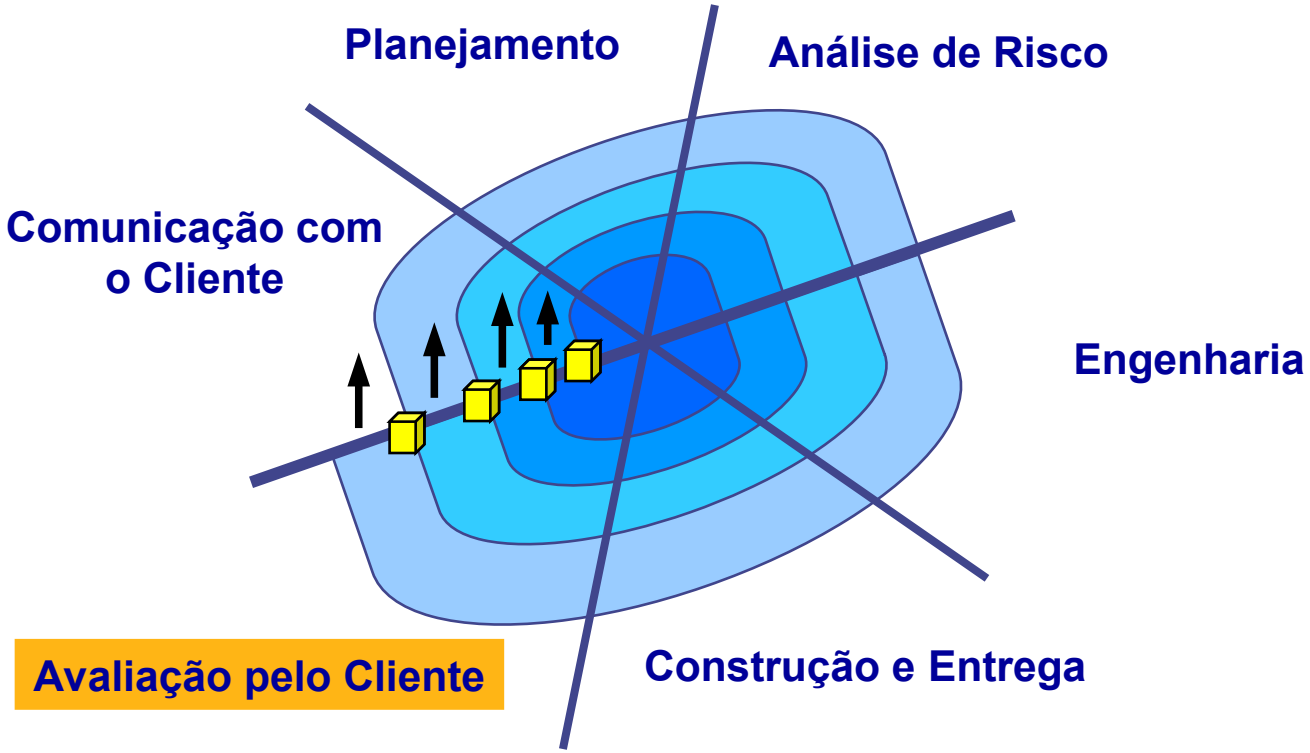
Modelo Espiral

- Tarefas requeridas para construir uma versão funcional ou próxima do funcional (protótipo avançado) da aplicação.
- Engloba atividades de Projeto e Codificação)



Modelo Espiral

- Tarefas requeridas para avaliar a versão corrente da aplicação de acordo com as necessidades do cliente.



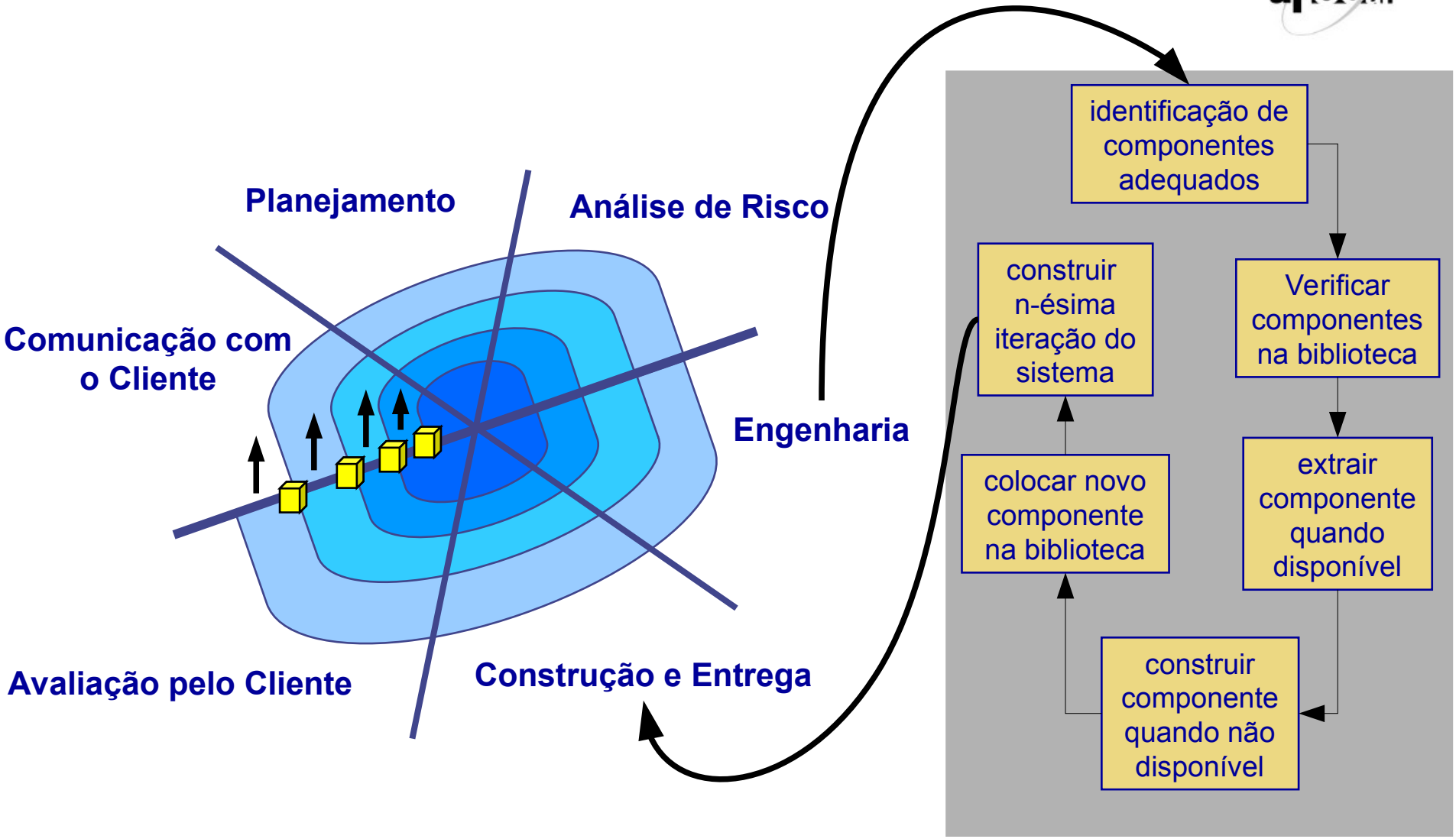
Comentários sobre o Modelo Espiral

- É uma abordagem realística para o desenvolvimento contemporâneo de software em grande escala.
- Usa uma abordagem que capacita o desenvolvedor e o cliente a entenderem e reagirem 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.

Leitura Complementar:
Modelo Baseado em Componentes

- Incorpora características de tecnologias Orientadas a Objetos no Modelo Espiral.
- A atividade de Engenharia começa com a identificação de componentes candidatos
  - Se o componente existe, ele será reutilizada
  - Se o componente não existe, ele será desenvolvida nos moldes do paradigma de Orientação a Objetos.

Leitura Complementar:
Modelo Baseado em Componentes



Leitura Complementar:
Modelo Baseado em Componentes

- Leva ao reúso de software.
- Estudos já indicaram que reúso traz uma redução de 70% nos prazos e 84% nos custos.
  - Índice de produtividade de 26,2 (padrão industrial é de 16,9);
  - Esses resultados são proporcionais à qualidade da biblioteca de componentes;
- Processo Unificado de desenvolvimento de software:
  - Define os componentes e as interfaces entre os componentes.
  - Define a função do sistema usando abordagem baseada em cenários.
  - Combina a função com uma arquitetura estrutural que identifica a forma que o software terá.

Roteiro

- Fases Genéricas de um Processo de Software.
- Modelos de Processo de Software
  - Modelo Cascata
  - Modelo RAD
  - Modelos Evolutivos
  - **Métodos Ágeis**

Métodos Ágeis

- Safra mais recente de formas de desenvolvimento de software resultante de métodos empíricos.
- Surgiram com o propósito de reduzir o *overhead* (sobrecarga) imposto por métodos rígidos ao desenvolvimento de sistemas corporativos de pequeno e médio porte.
- Aproximadamente 10 anos de "pesquisa de campo".
- Necessidade de entrega rápida e flexível.

Métodos Ágeis

- O **Manifesto Ágil** é um marco para o surgimento dos métodos ágeis:
  - <https://agilemanifesto.org/>

  - "Individuals and interactions over processes and tools.
  - Working software over comprehensive documentation.
  - Customer collaboration over contract negotiation.
  - Responding to change over following a plan."

Métodos Ágeis

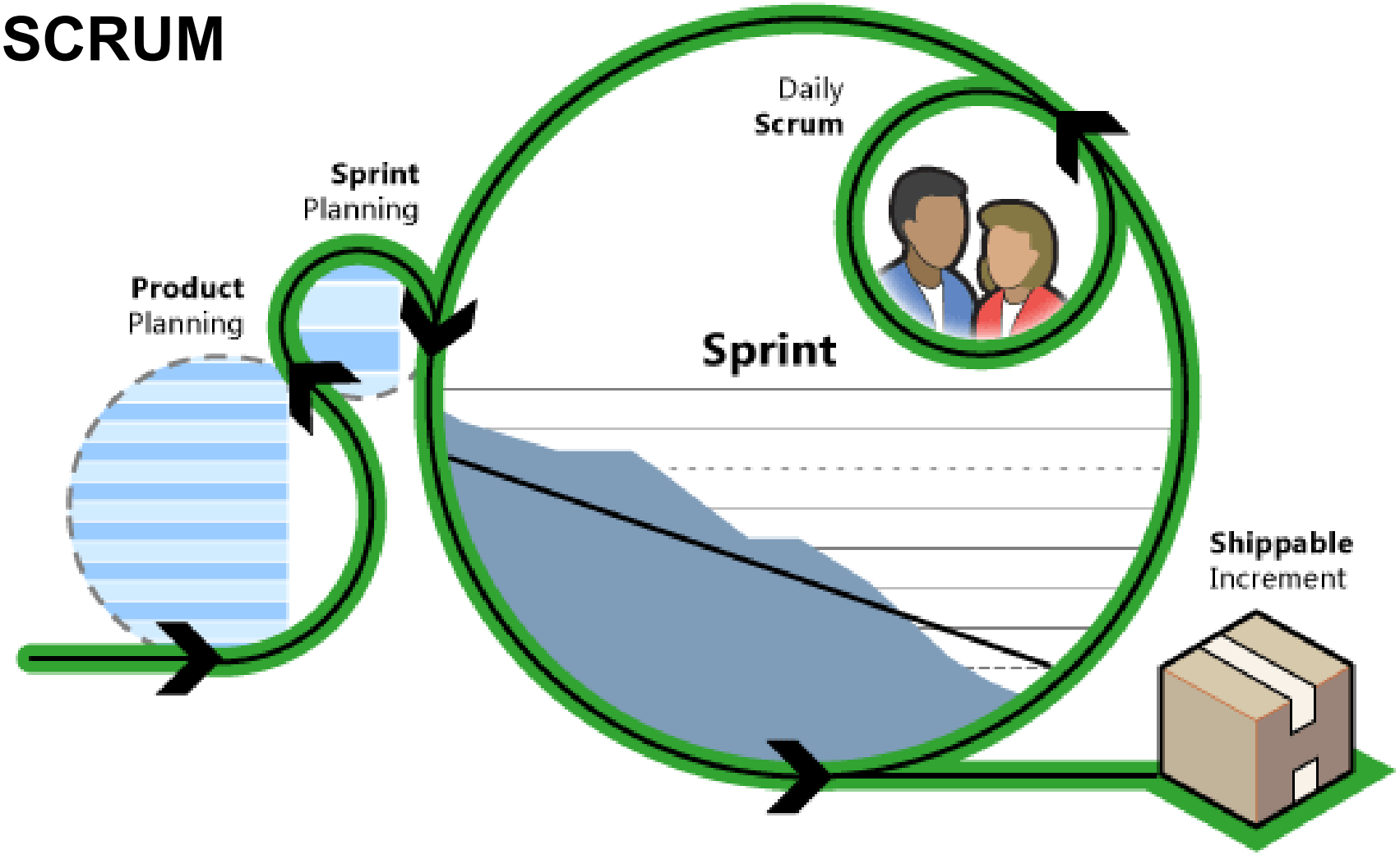
- Os mais conhecidos são:
  - eXtreme Programming (XP)
  - Feature-Driven Development (FDD)
  - Scrum
  - Crystal Methods
  - Open Source Development Process

Métodos Ágeis

- Características comuns:
  - Ciclos curtos de desenvolvimento
  - Planejamento para a iteração corrente
  - Times co-localizados
  - Testes de unidade
  - Controle de versões
  - Compilação automática de código
  - ...

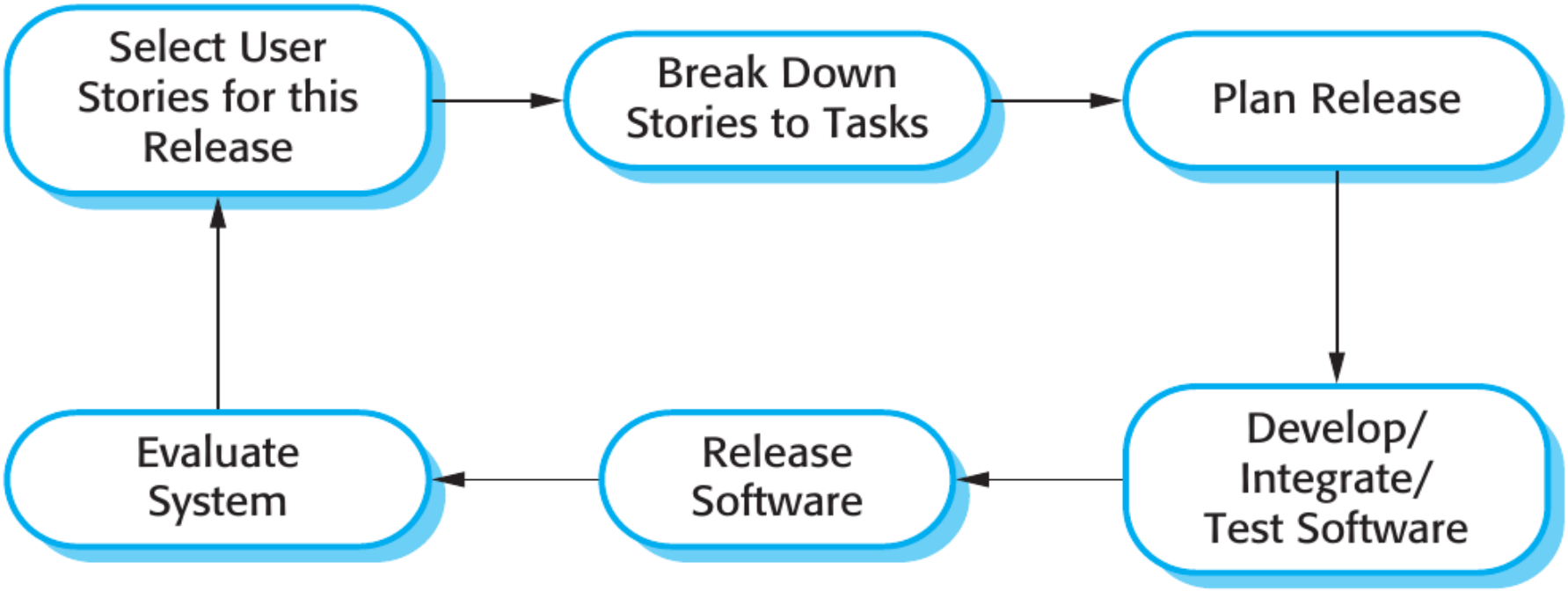
Métodos Ágeis: Exemplos

SCRUM



Métodos Ágeis: Exemplos

XP



Notas Adicionais

- Atualmente, qual é o ¨carro chefe¨?
  - **Continuous integration**: "*software development practice where members of a team integrate their work frequently, usually each person integrates at least daily - leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible. Many teams find that this approach leads to significantly reduced integration problems and allows a team to develop cohesive software more rapidly.*"

Fonte: <https://martinfowler.com/articles/continuousIntegration.html>

Próxima aula...

- Engenharia de Requisitos de Software