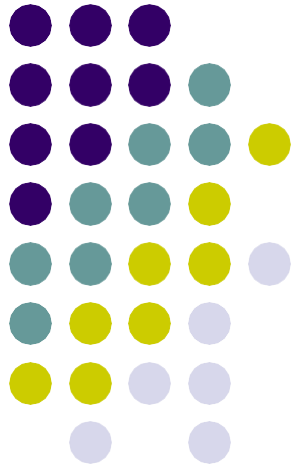
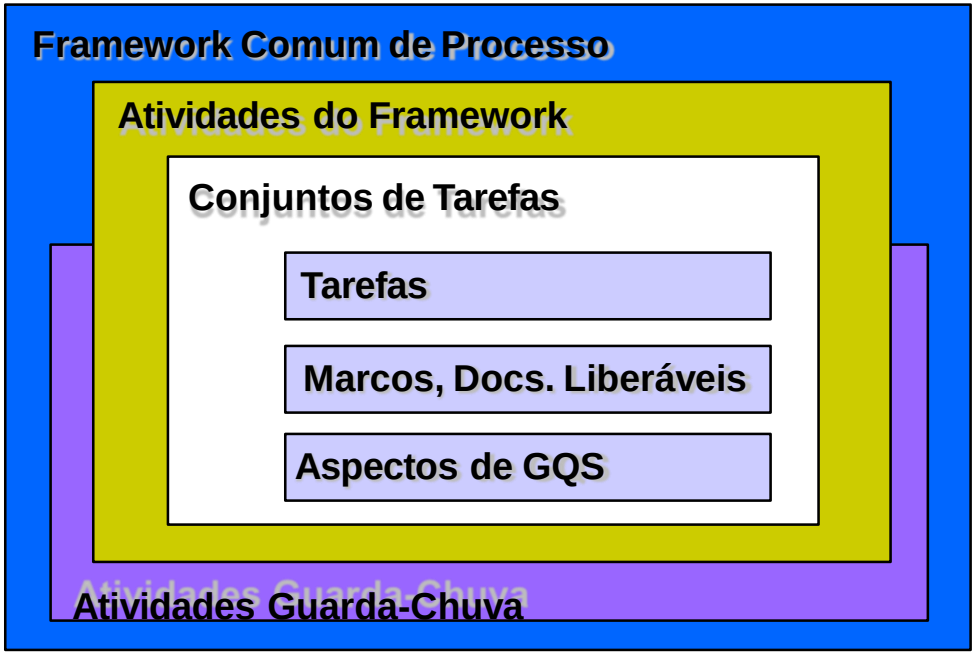
Engenharia de Software I



Prof. Ricardo Satoshi

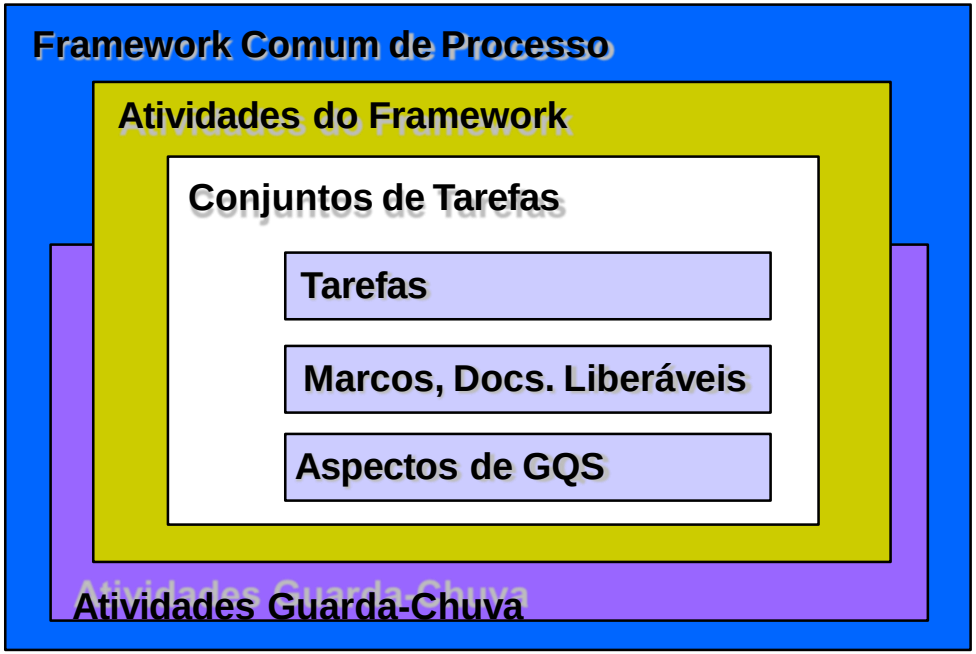
O Processo de Software



Framework Comum de Processo:

é estabelecido através da definição de um pequeno número de atividades que são aplicáveis a qualquer projeto de software, independentemente de seu tamanho ou complexidade

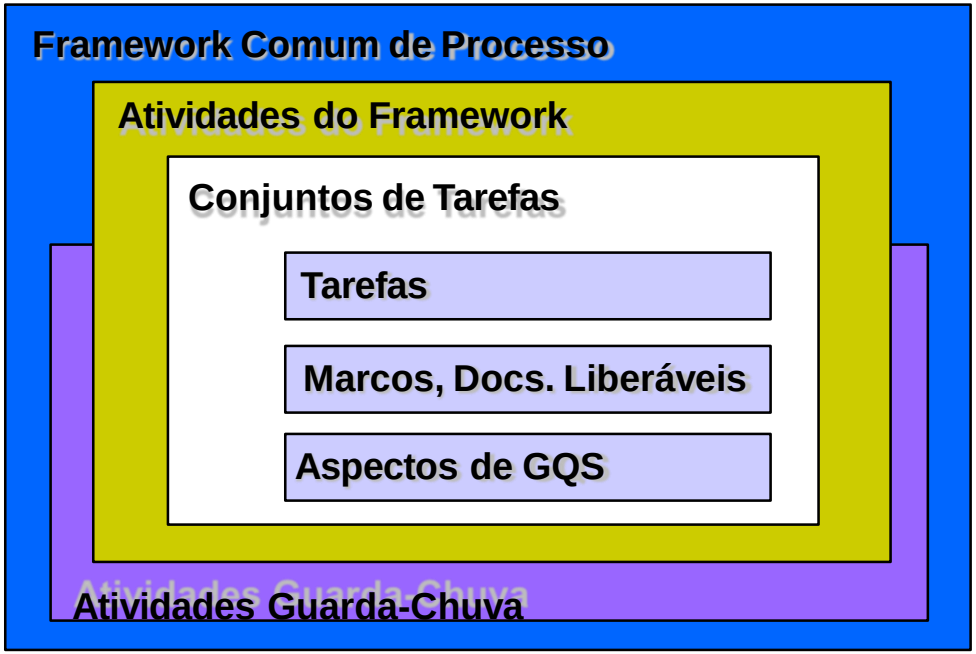
O Processo de Software



Conjuntos de Tarefas:

Coleção de tarefas da Eng. Software - marcos de projeto, produtos de software, docs. liberáveis e pontos de garantia de qualidade - que permite que o framework de atividades seja adaptado às características do projeto e à equipe

O Processo de Software



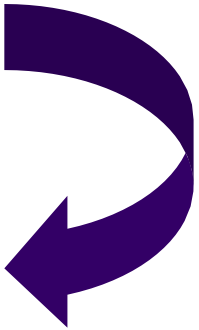
Atividades Guarda-Chuva:

Cobrem todo o processo e são independentes do framework de atividades

Modelos de Processo de Software

- processos
- métodos
- ferramentas
- fases genéricas

devem incorporar uma *estratégia* de desenvolvimento



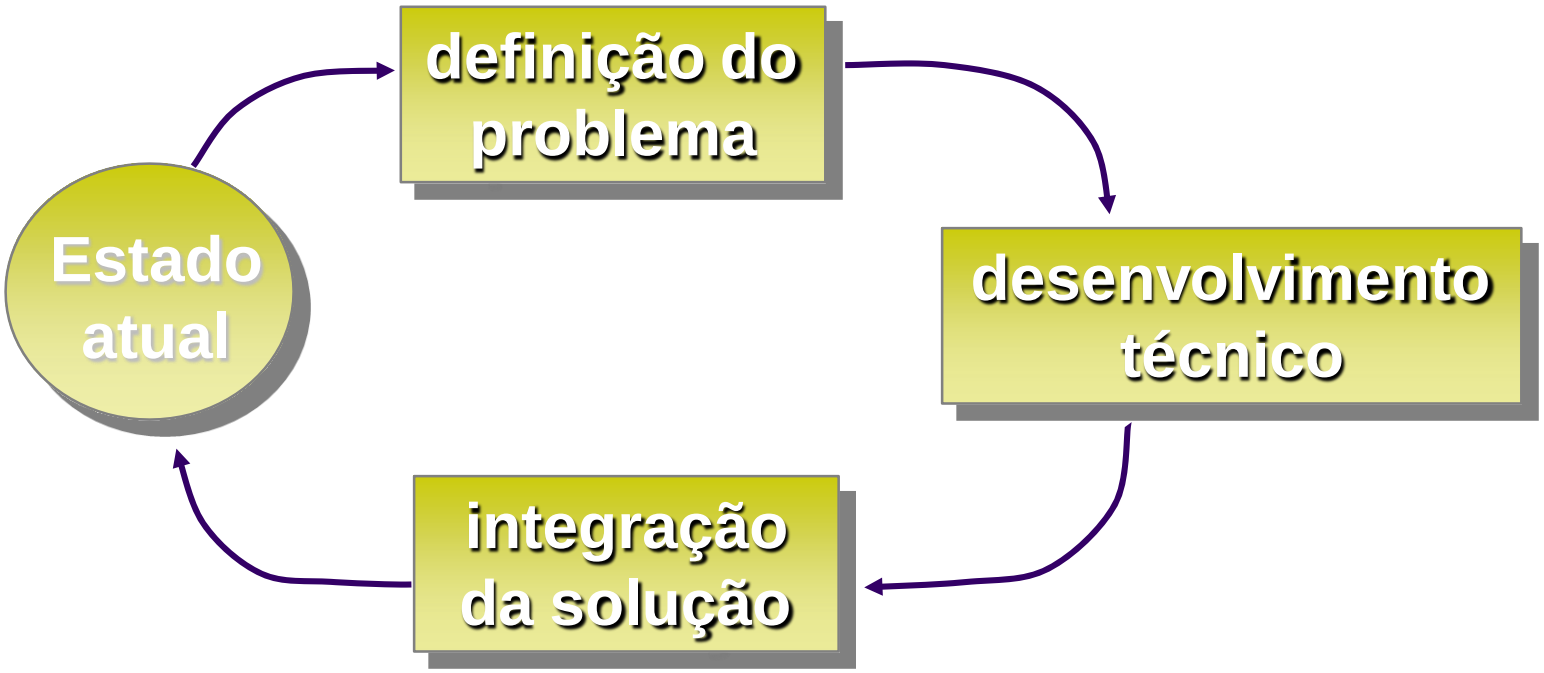
Modelo de Processo ?
Paradigma de Engenharia de Software ?
Ciclo de Vida ?

Modelo de Processo de Software

- 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

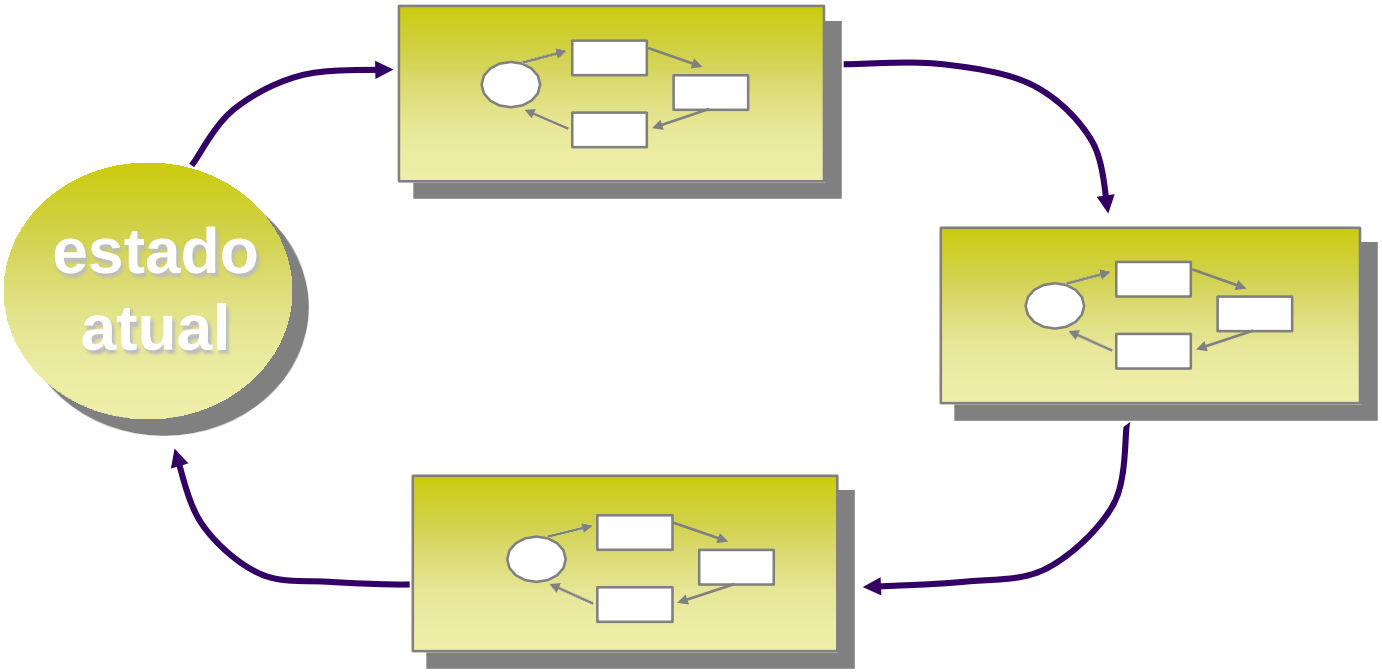
Modelo de Processo de Software

Todo desenvolvimento de software pode ser caracterizado como um loop de solução



Modelo de Processo de Software

O loop de solução pode ser usado em diferentes níveis de resolução



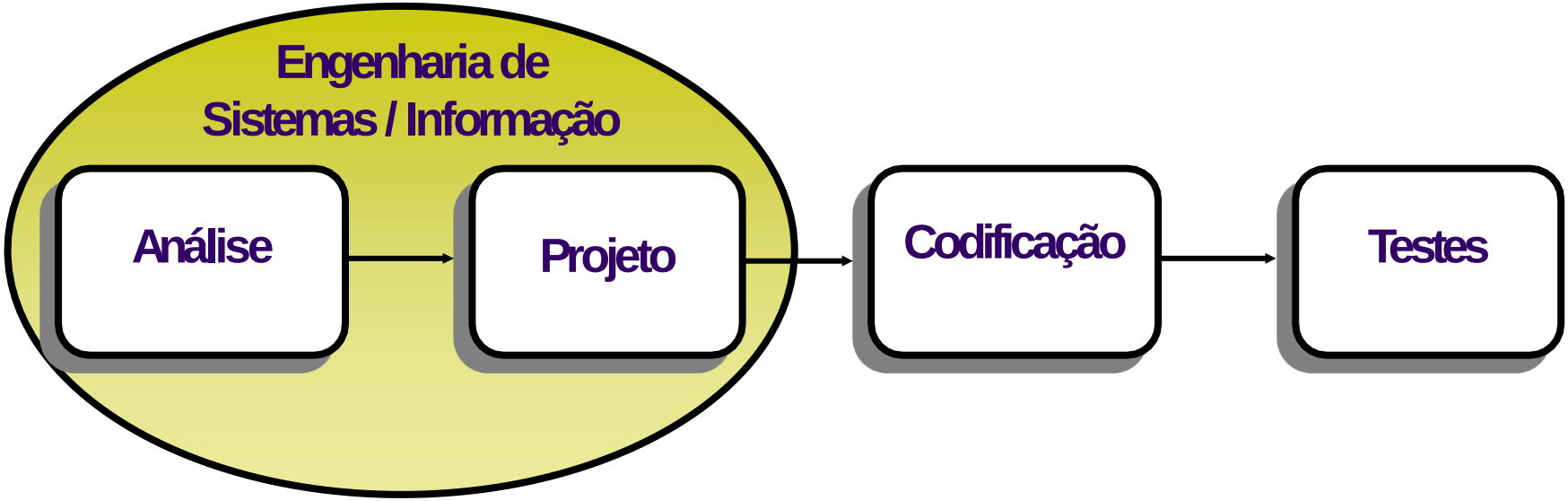
Modelos de Processo de Software

- Alguns Modelos de Processo:
  - Modelo Seqüencial Linear
  - Modelo de Prototipação
  - Modelo RAD
  - Modelos Evolucionários
    - Incremental
    - Espiral
    - Montagem de Componente
    - Desenvolvimento Concorrente
  - Modelo de Métodos Formais
  - Técnicas de 4a Geração

Modelo Sequencial Linear

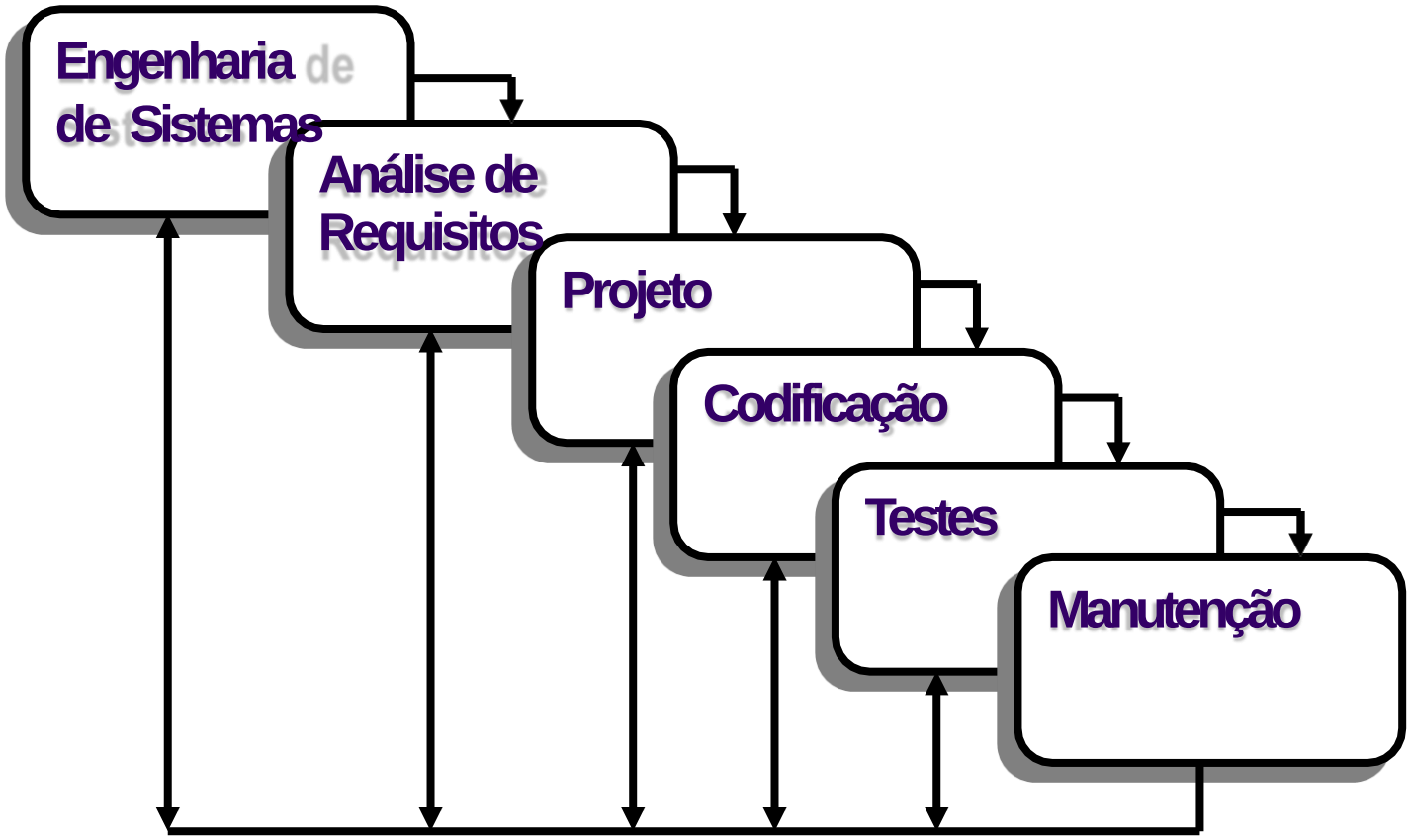
- Ciclo de Vida Clássico ou Modelo Cascata
- Modelo mais antigo e o mais amplamente usado da engenharia de software
- Modelado em função do ciclo da engenharia convencional
- Requer uma abordagem sistemática, seqüencial ao desenvolvimento de software
- o resultado de uma fase se constitui na entrada da outra

Modelo Sequencial Linear



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

Modelo Seqüencial Linear



modelo original, proposto por Royce, prevê feedback

Atividades do Modelo Sequencial Linear

- **Análise e Engenharia de Sistemas**
  - 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 Seqüencial Linear

- **Análise de Requisitos de Software**
  - 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 Seqüencial Linear

- **Projeto**
  - Tradução dos requisitos do software para um conjunto de representações que podem ser avaliadas quanto à qualidade, antes que a codificação se inicie
  - Se concentra em 4 atributos do programa:
    - Estrutura de Dados,
    - Arquitetura de Software,
    - Detalhes Procedimentais e
    - Caracterização de Interfaces

Atividades do Modelo Sequencial Linear

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

Atividades do Modelo Sequencial Linear

- **Testes**
  - Concentram-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.

Atividades do Modelo Seqüencial Linear

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

Problemas com o Modelo Sequencial Linear

- Projetos reais raramente seguem o fluxo sequencial 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
- Muitas vezes os desenvolvedores ficam ociosos desnecessariamente, devido a estados bloqueadores (quando existem tarefas dependentes, membros da equipe devem aguardar que outros terminem)

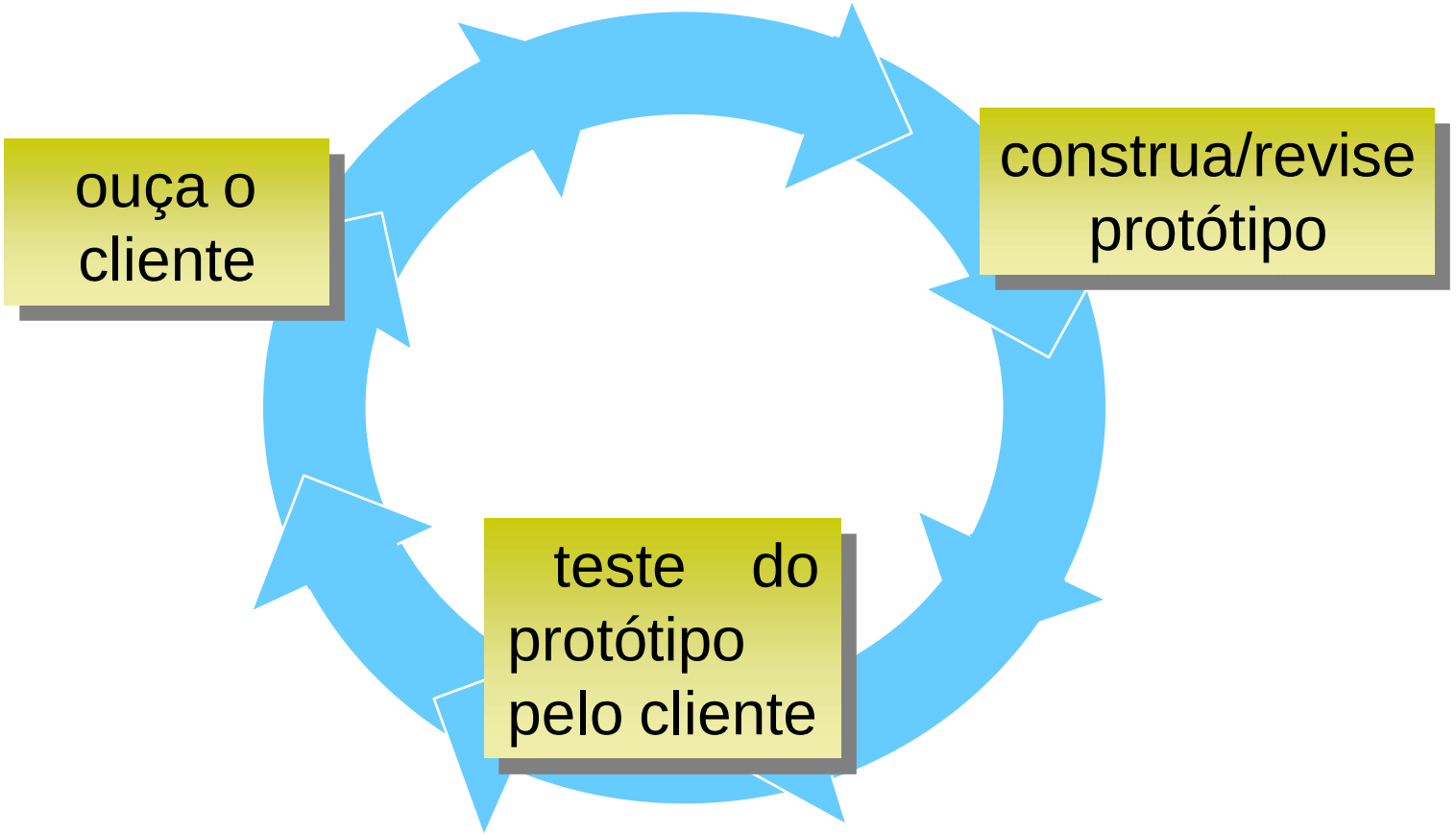
Modelo Sequencial Linear (comentários)

- Embora o Modelo Sequencial Linear ou Ciclo de Vida Clássico tenha fragilidades, ele é significativamente melhor do que uma abordagem casual ao desenvolvimento de software
- 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

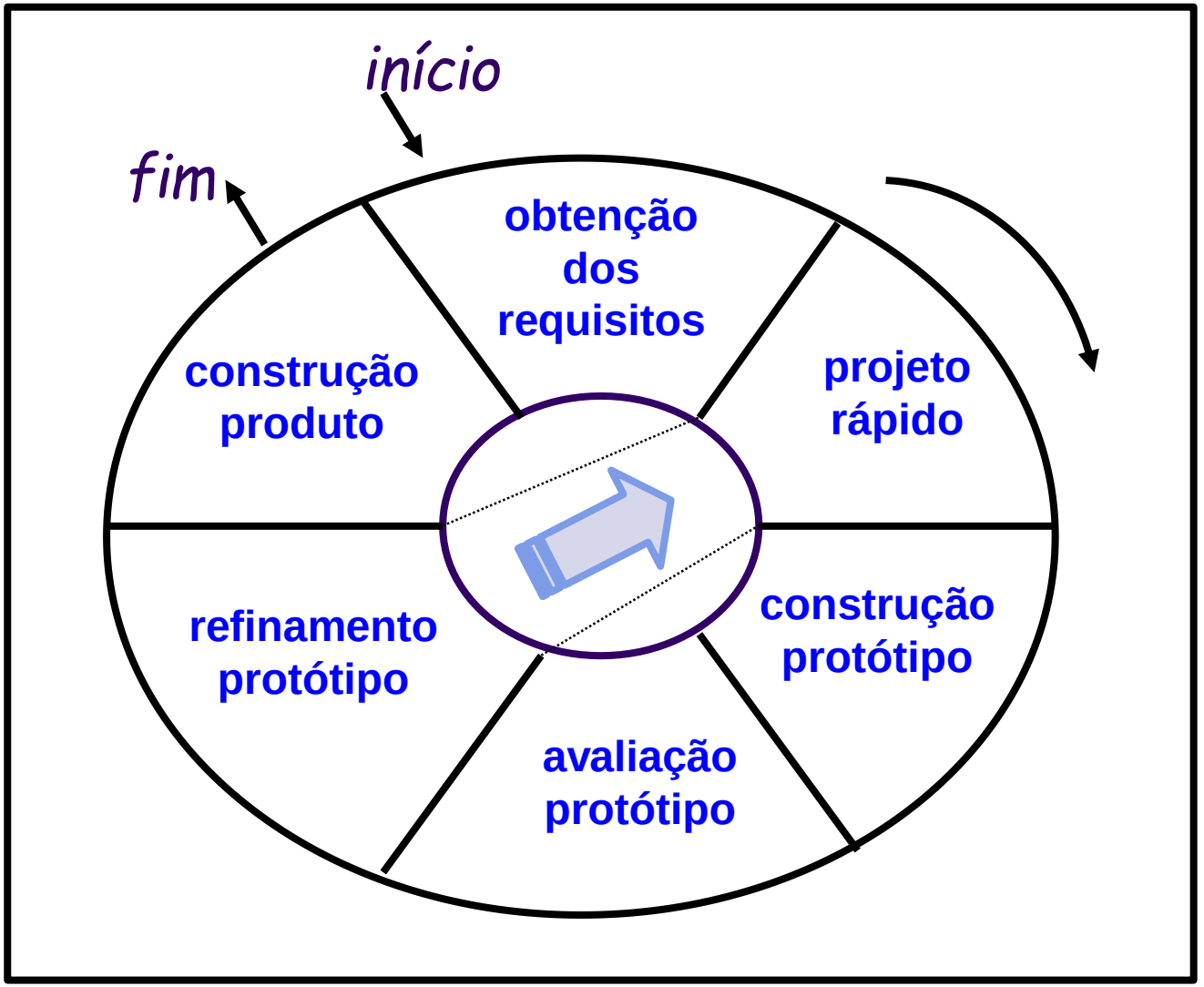
Prototipação

- 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.
- Apropriado para quando o cliente definiu um conjunto de objetivos gerais para o software, mas não identificou requisitos de entrada, processamento e saída com detalhes.

Prototipação



Prototipação



Atividades da Prototipação

Obtenção dos Requisitos

- Desenvolvedor e cliente definem os objetivos gerais do software, identificam quais requisitos são conhecidos e as áreas que necessitam de definições adicionais

Projeto Rápido

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

Atividades da Prototipação

Construção Protótipo

- Implementação rápida do projeto

Avaliação do Protótipo

- Cliente e desenvolvedor avaliam o protótipo

Atividades da Prototipação

- **Refinamento dos Requisitos**
  - cliente e desenvolvedor refinam os requisitos do software a ser desenvolvido.
- 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.

Atividades da Prototipação

- **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 (cliente nem sabe o que é isso!!!).
- Não aceita bem a idéia de que a versão final do software vai ser construída e "força" a utilização do protótipo como produto final.

Problemas com a Prototipação

- Desenvolvedor freqüentemente 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.
- O Protótipo vira o Produto.

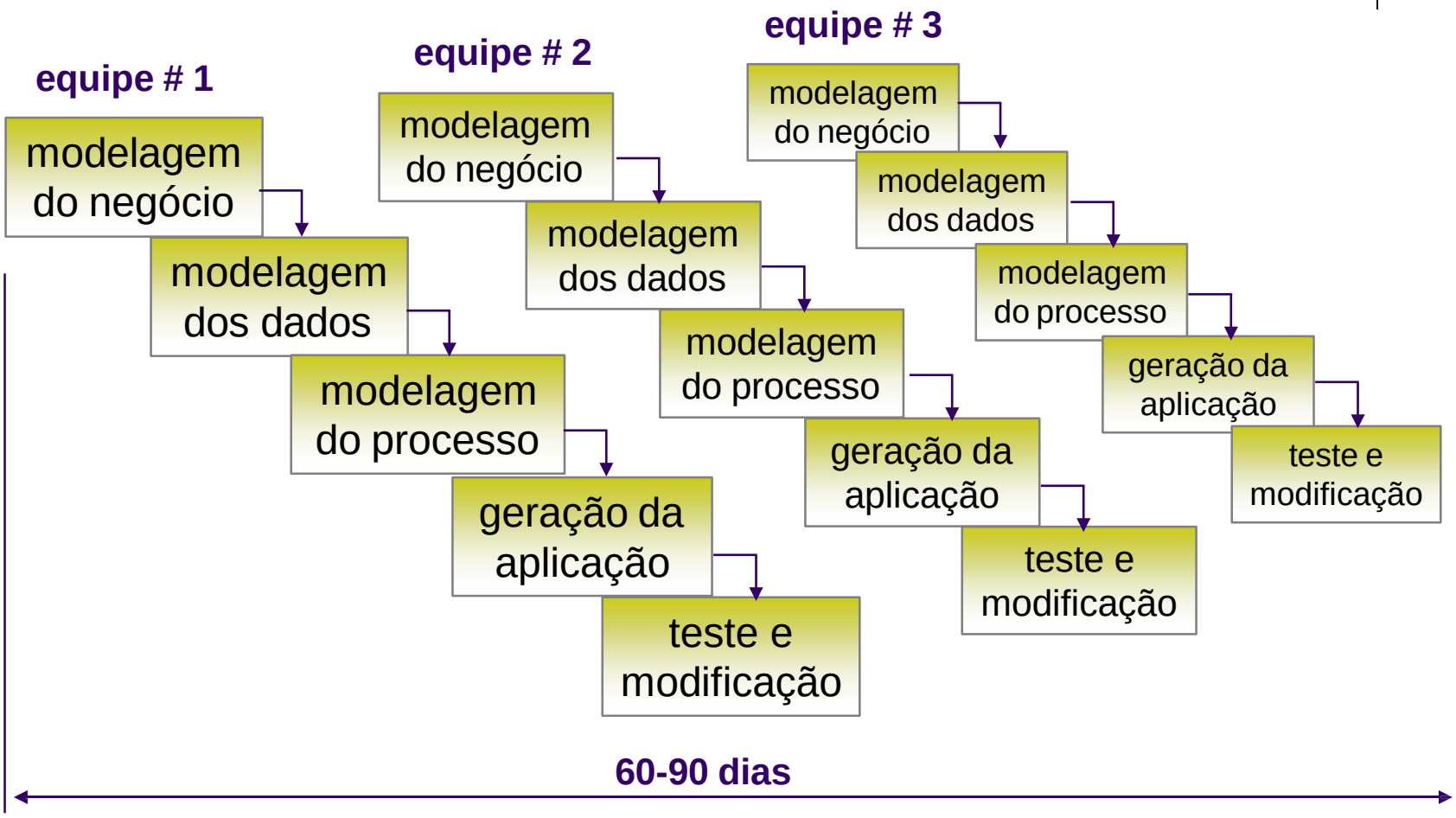
Prototipação (comentários)

- Ainda que possam ocorrer problemas, a prototipação é um ciclo de vida eficiente ✓
- A chave: definirem-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 ✓

Modelo RAD
(Rapid Application Development)

- É o modelo seqüencial linear mas 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

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
60-90 dias

Atividades do Modelo RAD

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

Atividades do Modelo RAD

- **Modelagem dos dados**
  - 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

Atividades do Modelo RAD

- **Modelagem do processo**
  - 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

Atividades do Modelo RAD

- **Geração da aplicação**
  - O modelo RAD assume o uso de técnicas de 4a. geração; ao invés de criar software de forma convencional, reusa componentes quando possível ou cria componentes reutilizáveis;
  - Ferramentas automatizadas são utilizadas para gerar software (GENECSUS)

Atividades do Modelo RAD

- **Teste e modificação**
  - 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

Modelo RAD

- Quando usar?
  - as restrições de tempo são impostas pelo projeto
  - quando 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 e, depois são integradas para formar o todo

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

Modelos de Processo Evolucionários

- Usado quando o deadline não é adequado para o desenvolvimento do software; a data de término não é realística
- Uma versão limitada (reduzida) pode ser introduzida para atender à competitividade e pressões do negócio
- São liberados “produtos beta”
- Os detalhes e extensões ainda devem ser definidos
  - Ex: editor de texto

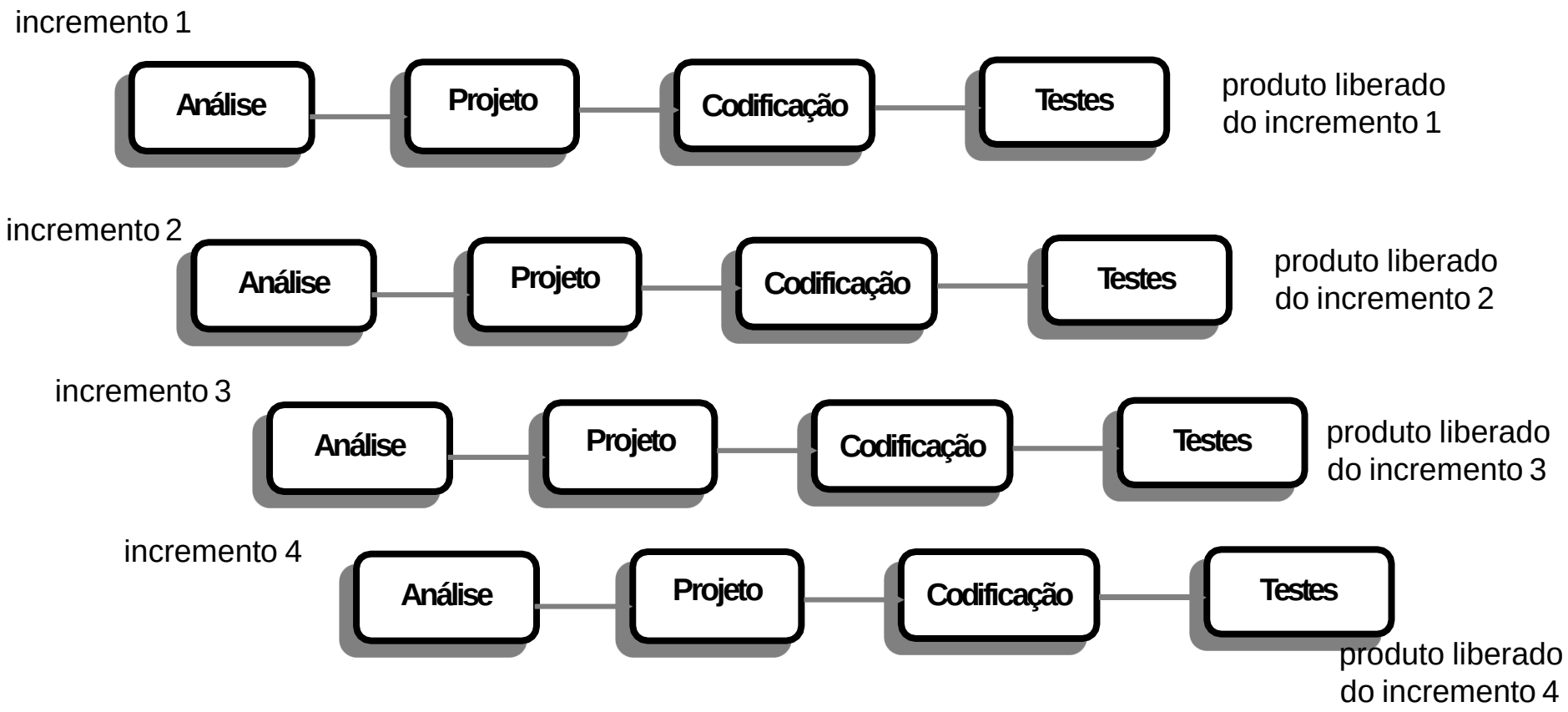
Modelos de Processo Evolucionários

- **Incremental**
- **Espiral**
- **Montagem de Componentes**
- **Técnicas de 4ª Geração**

Modelo Incremental

- combina elementos do Modelo Linear com a filosofia da Prototipação
- aplica sequências lineares numa abordagem de “saltos” à medida que o tempo progride
- O processo se repete até que um produto completo seja produzido
- Difere da Prototipação pois a cada incremento produz uma versão operacional do software

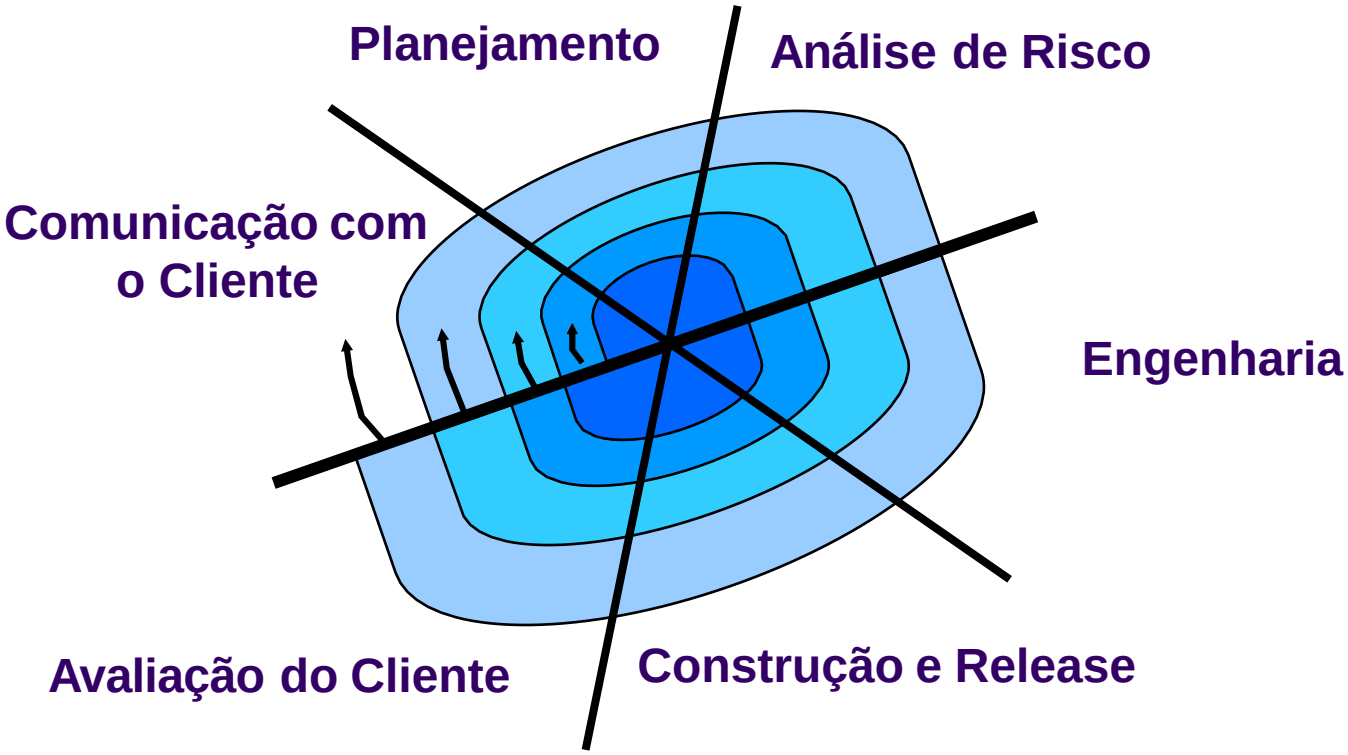
Modelo Incremental



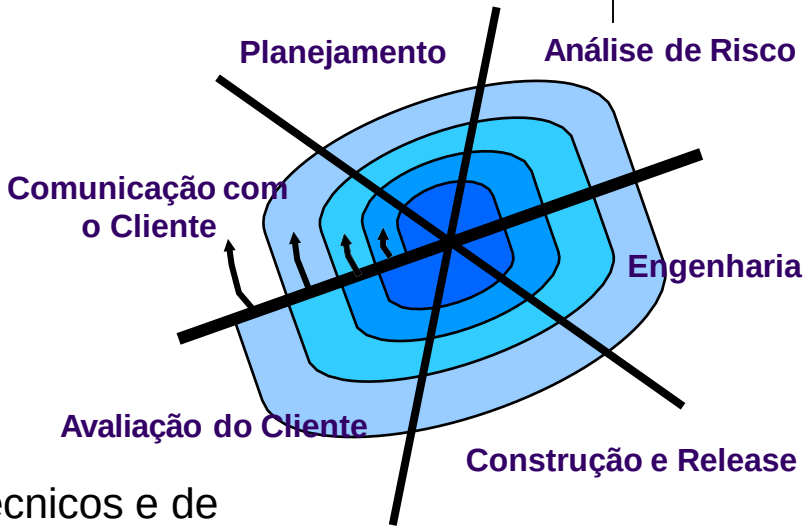
Modelo Espiral

- Engloba a natureza iterativa da Prototipação com os aspectos sistemáticos e controlados do Modelo Linear
- 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 andiantadas são produzidas versões incrementais mais completas e melhoradas

Modelo Espiral



Atividades do Modelo Espiral



Comunicação com o cliente:
tarefas requeridas para estabelecer uma efetiva comunicação entre desenvolvedor e cliente

Planejamento:
tarefas requeridas para definir recursos, referenciais de tempo e outras informações de projeto

Análise de Risco:
tarefas requeridas para fazer levantamento de riscos técnicos e de gerenciamento

Engenharia:
tarefas requeridas para construir uma ou mais representações da aplicação

Construção e Release:
tarefas requeridas para construir, testar, instalar e dar suporte ao usuário (p.ex., documentação e treinamento)

Avaliação do cliente:
tarefas requeridas para obter um feedback do cliente baseado na avaliação da representação do software criado durante a fase de engenharia e implementado durante a fase de instalação

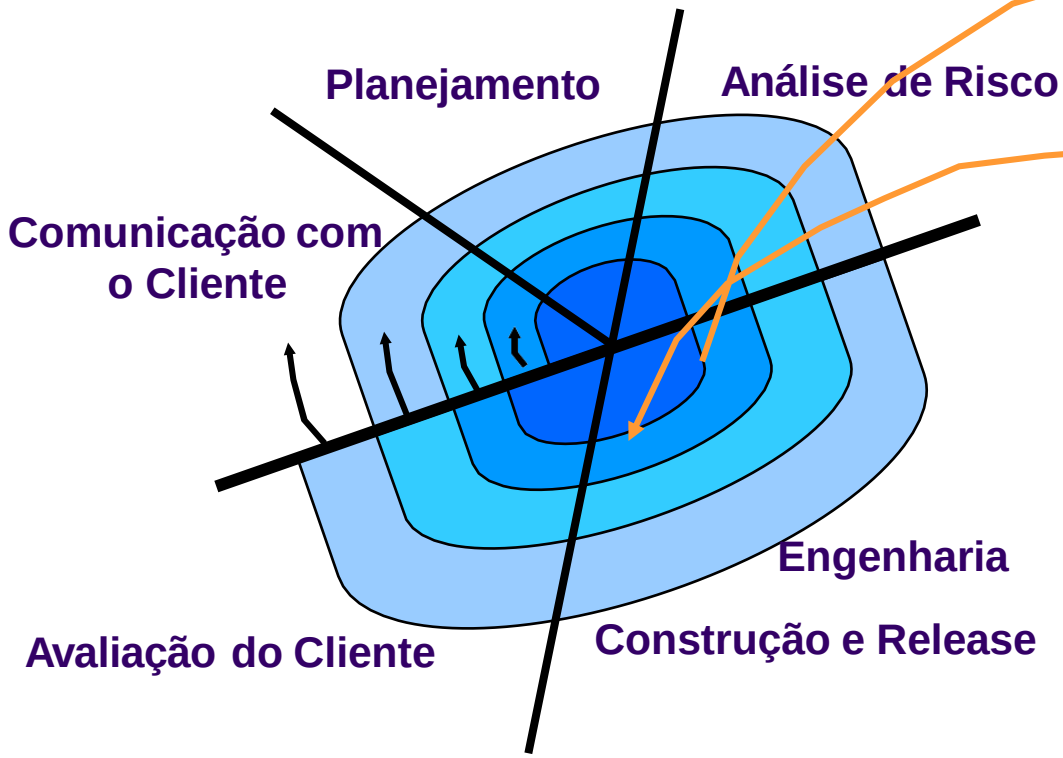
Modelo Espiral (comentários)

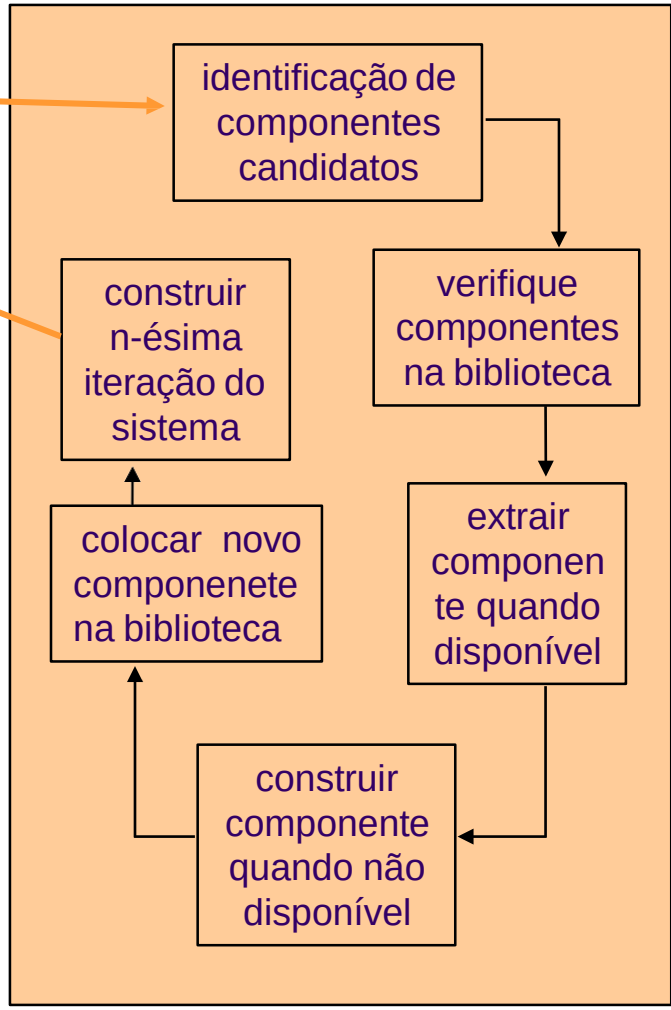
- É, atualmente, a abordagem mais realística para o desenvolvimento de software em grande escala.
- Usa uma abordagem que capacita o desenvolvedor e o cliente a entender e reagir aos riscos em cada etapa evolutiva.
- Pode ser difícil convencer os clientes que uma abordagem "evolutiva" é controlável.
- Exige considerável experiência na determinação de riscos e depende dessa experiência para ter sucesso.

Modelo de Montagem de Componentes

- Incorpora características de tecnologias Orientadas a Objetos no Modelo Espiral
- A atividade de Engenharia do Sistema começa com a identificação de classes candidatas
- Se a classe existe, ela será reutilizada
- Se a classe não existe, ela será desenvolvida nos moldes do paradigma de Orientação a Objetos

Modelo de Montagem de Componentes

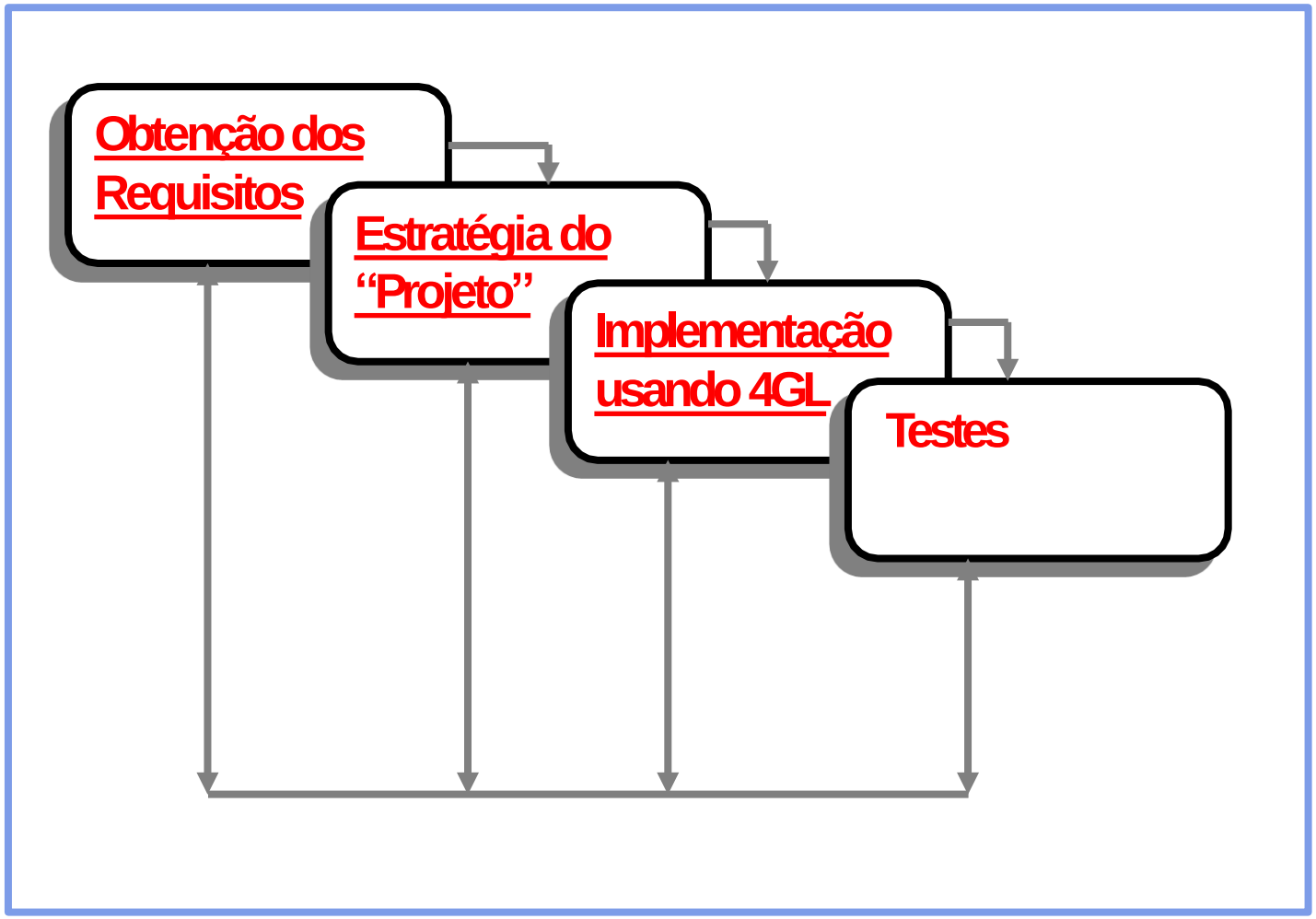




Técnicas de 4a Geração

- 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
  - que o código fonte seja gerado automaticamente a partir dessas especificações

Técnicas de 4a Geração



Ferramentas do ambiente de desenvolvimento de software de 4GL

- 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

Atividades das Técnicas de 4a Geração

1. 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 demorar para 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"

Atividades das Técnicas de 4a Geração

2. Estratégia de "Projeto":

- para pequenas aplicações é possível mover-se do passo de Obtenção dos Requisitos para o passo de Implementação usando uma Linguagem de 4G
- 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)

Atividades das Técnicas de 4a Geração

3. 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

Atividades das Técnicas de 4a Geração

4. Teste

- 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.

Técnicas de 4a Geração (comentários)

- 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

Conclusão

- 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]