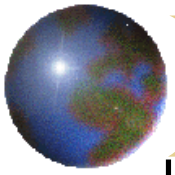
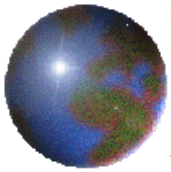


UNIVERSIDADE DE PASSO FUNDO

ENGENHARIA DE SOFTWARE
Modelos de Processos de desenvolvimento de Software

Prof. Jeangrei Veiga

jeangrei@upf.br



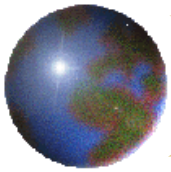
Modelo X Processo

- **Processo é um conjunto estruturado de atividades necessárias para desenvolver um sistema de software**
- **O processo deve determinar ações práticas a serem realizadas pela equipe, tais como prazos definidos e métricas para se avaliar como elas estão sendo realizadas**

Modelo X Processo

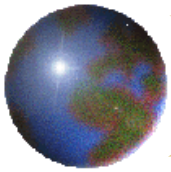
- **Um modelo é algo teórico, um conjunto de possíveis ações.**
- **Um modelo de processo de software é uma representação abstrata do processo.**

Modelo + Planejamento = Processo

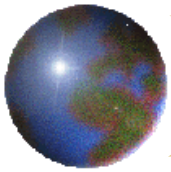


Modelo de Processo

- **Um modelo de processo de software deve ser 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;



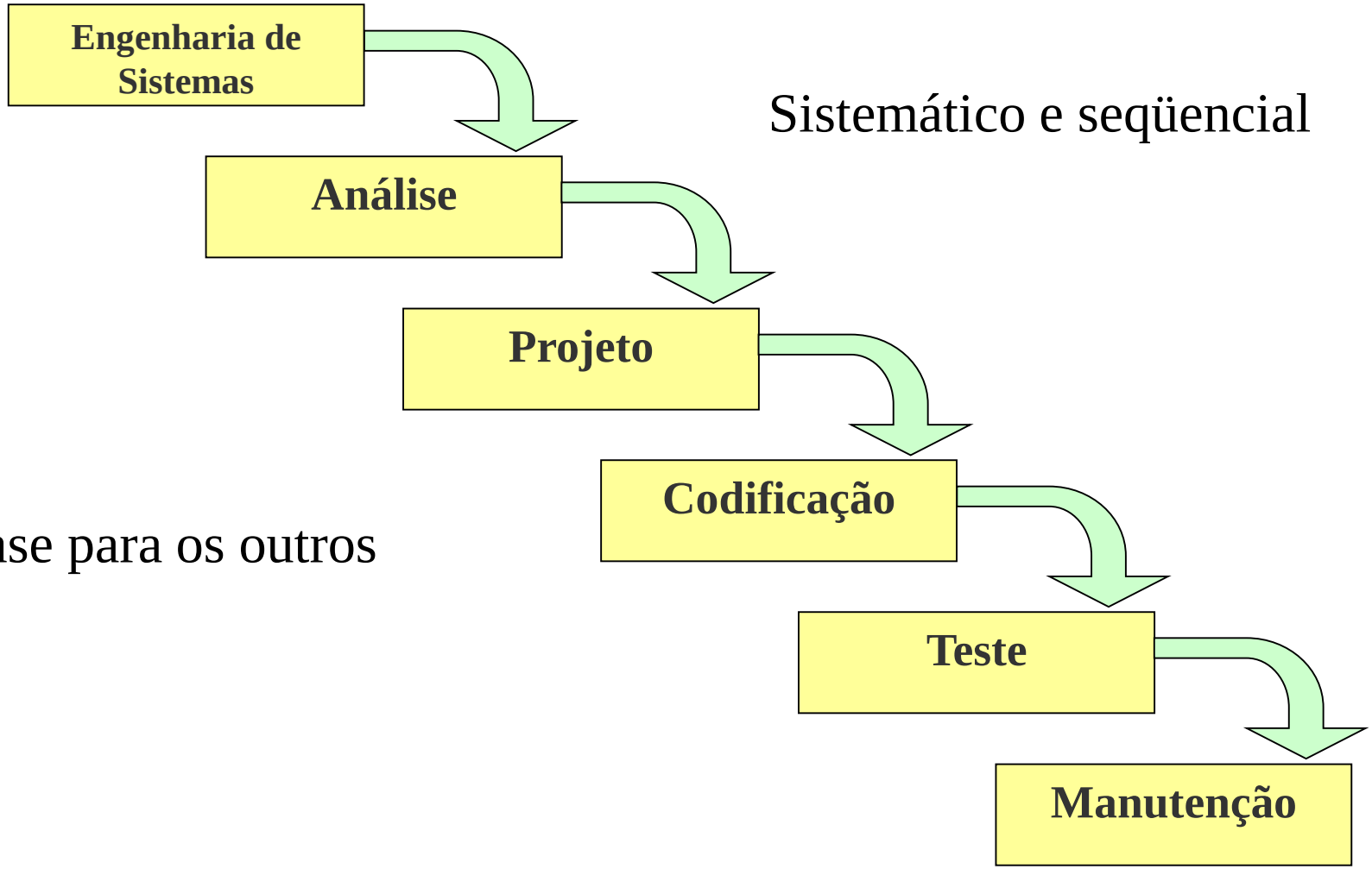
Modelo Cascata



Modelo em Cascata

- **Um dos primeiros modelos (Royce, 1970).**
- **O desenvolvimento de um estágio deve terminar antes do próximo começar.**
- **Simples, mas não reflete, efetivamente, o modo como o código é desenvolvido.**
- **Derivado do mundo do hardware (linhas de montagens).**

Modelo em Cascata



Sistemático e seqüencial

Base para os outros

Modelo em Cascata

Engenharia de Sistemas

- **Software faz parte de um sistema maior;**
- **Estabelecer os requisitos básicos para todos os elementos que envolvem o software, como hardware, pessoas e bancos de dados.**
- **Envolve a coleta dos requisitos em nível do sistema, com uma pequena quantidade de projeto e análise de alto nível.**
- **Exige uma intensa comunicação entre o cliente e o analista**
- **Faz parte da Analise de Sistema**

Modelo em Cascata

Análise dos Requisitos

- **Intensifica-se o processo de coleta dos requisitos;**
- **Identificar as funções necessárias, o desempenho e interfaces exigidos. (funcionalidades e restrições)**
- **Os requisitos para o sistema e para o software são documentados e revistos com o cliente.**
- **Produz a especificação dos requisitos.**
- **Faz parte da Analise de Sistema.**

Modelo em Cascata

Projeto

- **Traduz os requisitos em um conjunto de representações que podem ser avaliadas quando à qualidade.**
  - Estrutura de dados;
  - Arquitetura do software;
  - Detalhes Procedimentais;
  - Caracterização da interface.

- **É avaliado antes de começar a ser implementado;**

- **Junto com as etapas anteriores torna-se parte da documentação do sistema.**

Modelo em Cascata

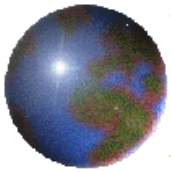
Codificação

- **Projeto traduzido para a linguagem do computador(C, Delphi, Java).**
- **Se o projeto for executado detalhadamente, a codificação pode ser executada mecanicamente?**

Modelo em Cascata

Testes

- **Concentra-se nos aspectos funcionais externos e lógicos internos do software.**
- **Garante que “todas as instruções” tenham sido testadas.**
- **A entrada definida produz os resultados exigidos?**



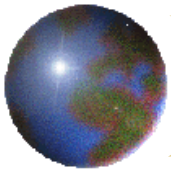
Modelo em Cascata

Manutenção

- **Software embutido nem sempre tem esta parte;**
- **provavelmente o software deverá sofrer mudanças depois que for entregue ao cliente**

Modelo em Cascata

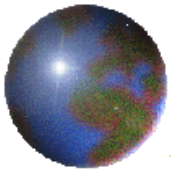
- **Tipos de Manutenção:**
  - Manutenção Corretiva: diagnóstico e correção de erros
  - Manutenção Adaptativa: adaptação do software para acomodar mudanças em seu ambiente externo.
  - Manutenção Perfectiva: exigência do cliente para acréscimos funcionais e de desempenho
  - Manutenção Preventiva: melhorar a confiabilidade e manutenibilidade futura (técnicas de engenharia reversa e reengenharia)



Modelo em Cascata

Problemas

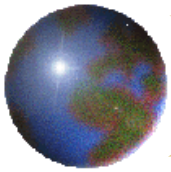
- **O mais antigo e amplamente usado.**
- **Projetos reais raramente seguem o fluxo seqüencial que ele propõe. Ocorrem iterações que trazem problemas na aplicação do paradigma.**
- **É difícil para o cliente declarar todas as exigências explicitamente. É difícil acomodar as incertezas naturais que existem no começo de muitos projetos.**



Modelo em Cascata

Problemas:

- **O cliente deve ter paciência. Uma versão do software só estará disponível em um ponto tardio do cronograma. Um erro crasso, pode ser desastroso.**
- **Desenvolvedores Ociosos.**
- **Só é apropriado quando os requisitos são bem conhecidos.**



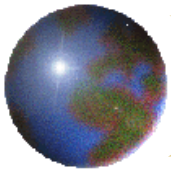
Prototipação

Prototipação

APROPRIADO QUANDO

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

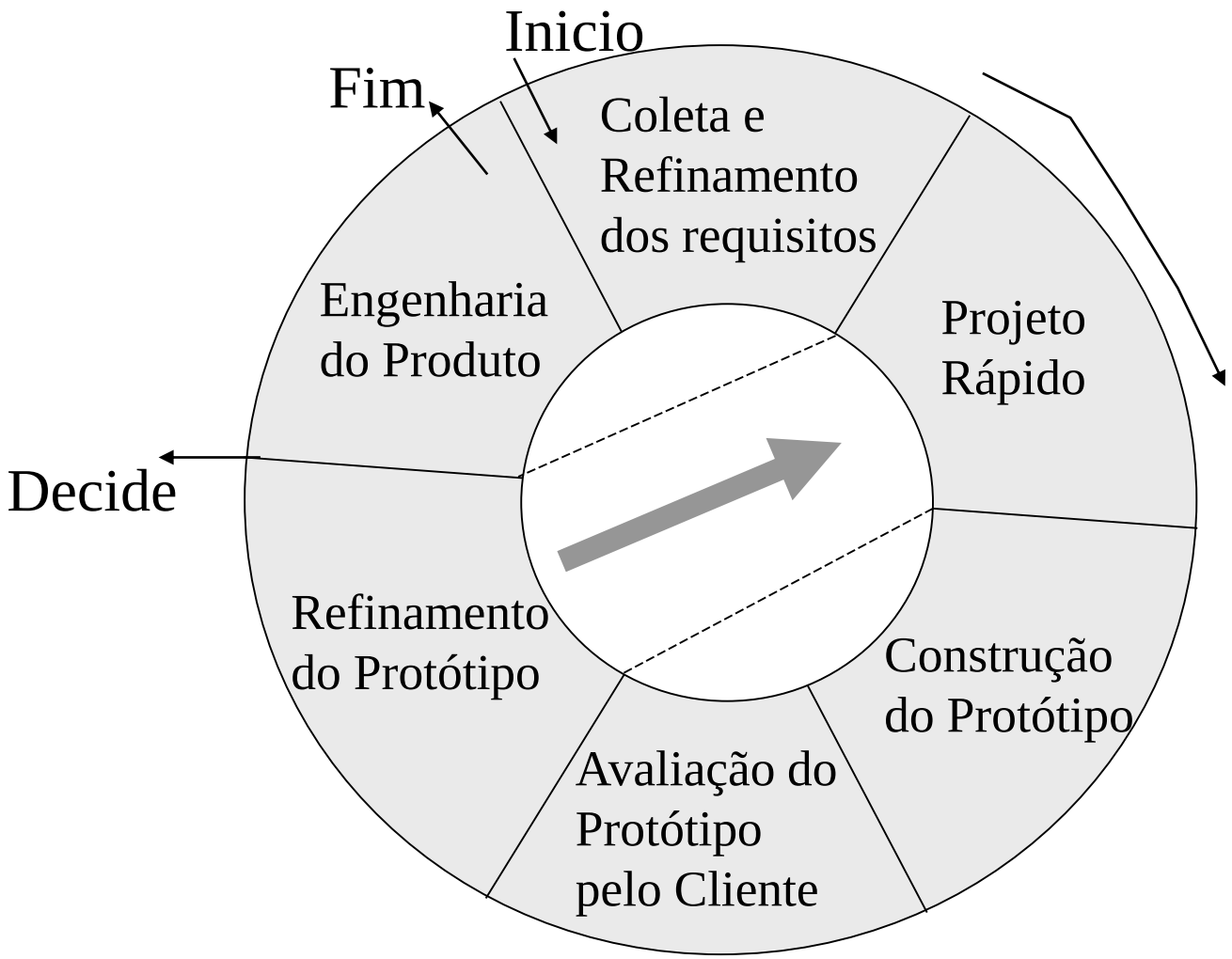
- **desenvolvedor não tem certeza da eficiência de um algoritmo, forma da interação homem/máquina.**

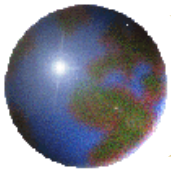


Prototipação

- **Permite o refinamento iterativo dos requisitos.**
- **A cada iteração é produzido um protótipo do software final.**
- **Este protótipo pode ser um:**
  - **Protótipo em Papel**, primeiras versões que permitem ao usuário ter uma visão abstrata do sistema;
  - **Protótipo incompleto**, implementa algum subconjunto de funções exigidas;
  - **Protótipo final**, um software que executa parte ou toda a função desejada, mas que tem outras características que serão melhoradas e ainda não pode ser disponibilizado.

Prototipação

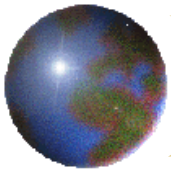




Prototipação

- **Coleta e Refinamento dos Requisitos:**
  - Nesta etapa o desenvolvedor e o cliente devem definir os objetivos gerais do software(Protótipo),
  - Identificar quais requisitos são conhecidos e as áreas que necessitam de definição adicional.
  - Análise de Sistema

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

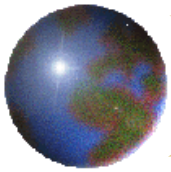


Prototipação

- **Construção do Protótipo:**
  - Implementação rápida do Projeto.

- **Avaliação do Protótipo:**
  - Cliente e desenvolvedor avaliam o protótipo.

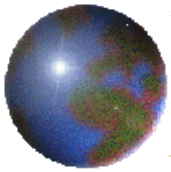
  - No caso de sugestão ou mudanças serão trabalhadas na próxima fase.



Prototipação

- **Refinamento do Protótipo:**
  - São trabalhados os problemas encontrados na fase anterior. Ou seja, são refinados os requisitos.
  - Neste ponto pode ocorrer, no caso de necessidade de alterações, um retorno na fase de projeto Rápido para desenvolver um novo protótipo que incorpora as mudanças.

- **Construção do Produto:**
  - Identificado todos os requisitos necessários, o protótipo pode ser descartado e a versão final do produto deve ser construída considerando os critérios de qualidade.



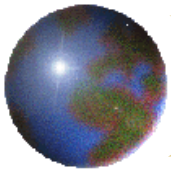
Prototipação

Problemas:

- **O cliente muitas vezes não aceita mais uma iteração, aquela versão mesmo incompleta já serve.**
- **Não há necessidade de desenvolver uma versão final, modifica-se o protótipo.**
- **desenvolvedor freqüentemente faz uma implementação comprometida (utilizando o que está disponível) com o objetivo de produzir rapidamente um protótipo.**

Solução:

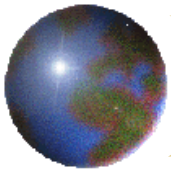
- **Definir as regras do jogo logo no começo, o cliente deve concordar que o protótipo seja construído para servir como um mecanismo a fim de definir os requisitos**



Modelo Espiral

Modelo em Espiral

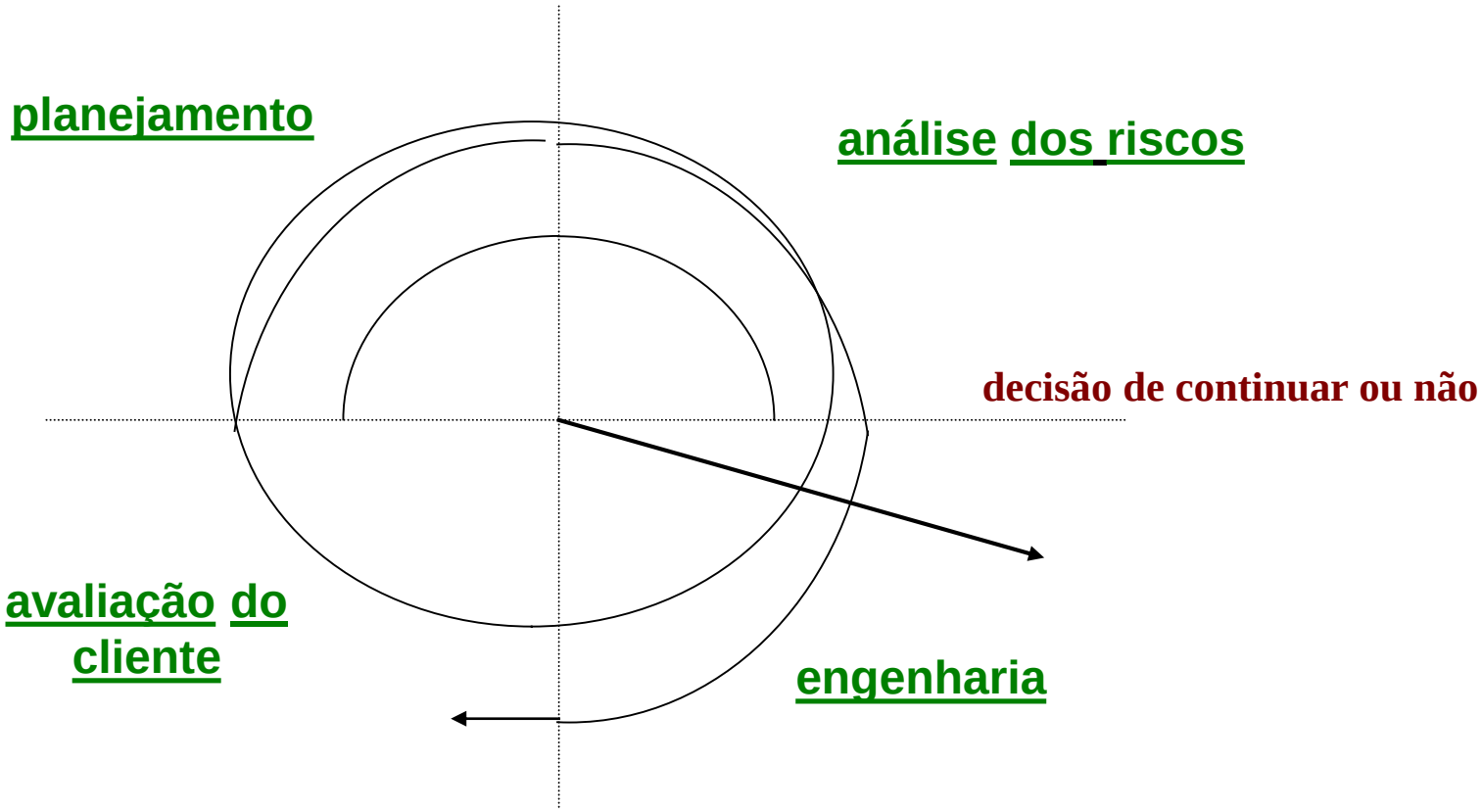
- **O Modelo em espiral é um modelo iterativo e incremental.**
- **Este tipo de modelo combina elementos do modelo cascata (aplicado repetidamente) com a filosofia iterativa da prototipação.**
- **Acrescenta mais uma atividade: Análise de Risco.**
- **O objetivo é trabalhar junto do usuário para descobrir seus requisitos, de maneira incremental, até que o produto final seja obtido.**

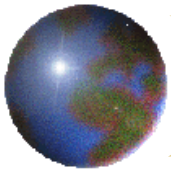


Modelo em Espiral

- **O desenvolvimento começa com as partes do produto que são mais bem entendidas.**
- **A evolução acontece quando novas características são adicionadas à medida que são sugeridas pelo usuário.**
- **A cada iteração é desenvolvido uma versão usável, não um protótipo.**

Modelo em Espiral





Modelo em Espiral

- **Fornece o potencial para desenvolvimento rápido de versões incrementais do software.**
- **Nas primeiras iterações são desenvolvido versões que podem ser protótipos.**
- **Nas iterações mais adiantadas são produzido versões incrementais mais completas e melhoradas.**

Modelo em Espiral

1- PLANEJAMENTO:

- determinação dos objetivos, alternativas e restrições;
- Comunicação com os clientes;
- Definição de Recursos.

2- ANÁLISE DE RISCO:

- análise das alternativas e identificação / resolução dos riscos

Modelo em Espiral

3- CONSTRUÇÃO:

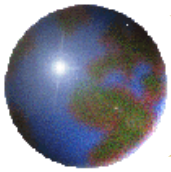
- desenvolvimento do produto no nível seguinte;
- Constrói protótipos ou versões mais avançadas do produto.
- Realiza Testes, implantação, suporte.

4- AVALIAÇÃO DO CLIENTE:

- Obter um feedBack do cliente baseado na avaliação da versão do software.
- São levantado as necessidades de mudança para o software

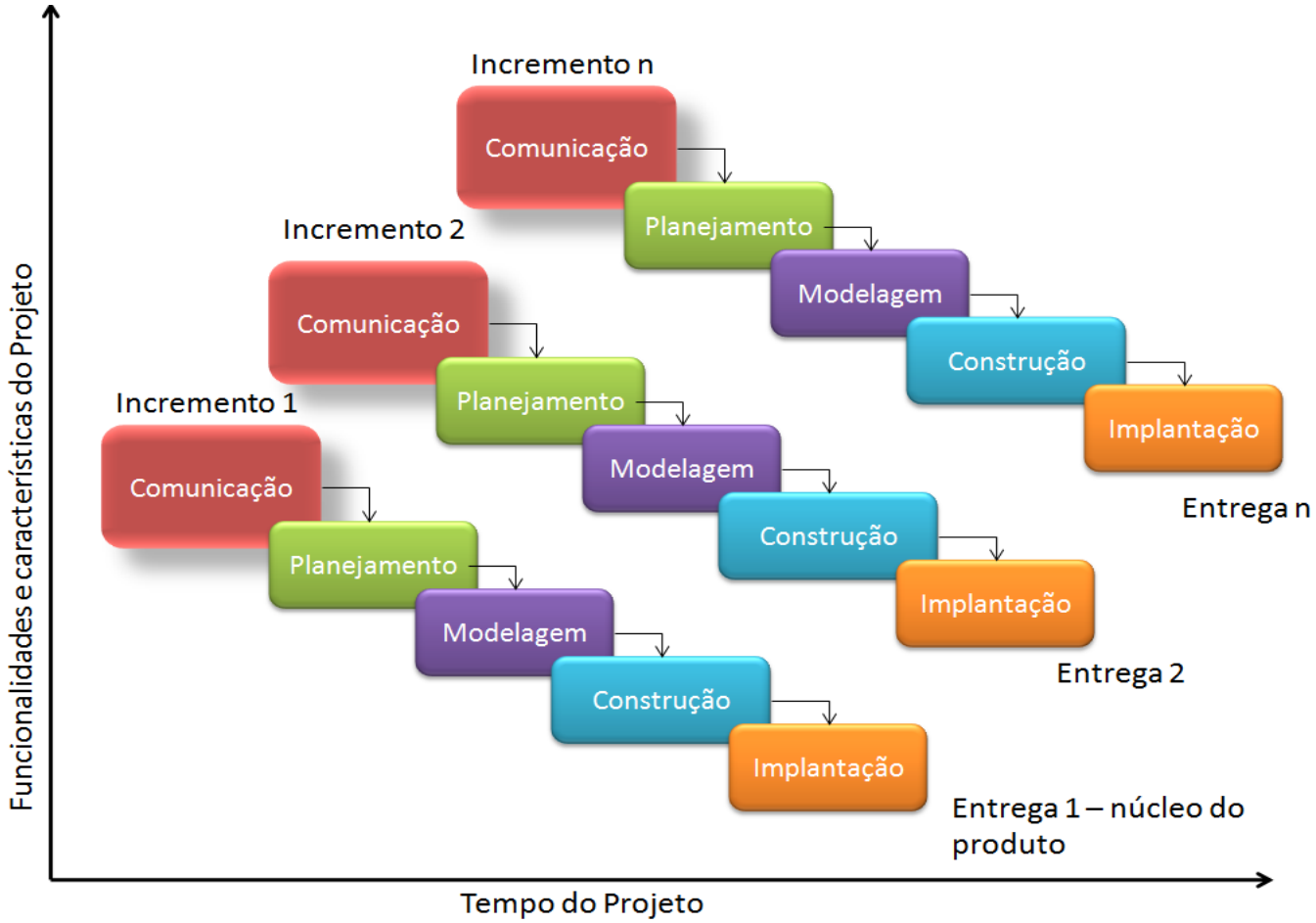
Modelo em Espiral

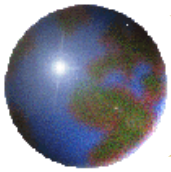
- **É uma abordagem realística para o desenvolvimento de software em grande escala;**
- **Usa uma abordagem que capacita o desenvolvedor e o cliente a entender e reagir aos riscos em cada etapa evolutiva;**
- **Pode ser difícil convencer os clientes que uma abordagem "evolutiva" é controlável;**
- **Exige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



Modelo Incremental

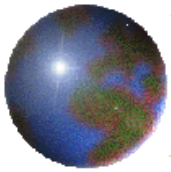
Modelo Incremental





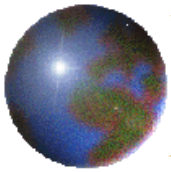
Modelo Incremental - Definição

Desenvolvimento Incremental é uma estratégia de planejamento estagiado em que várias partes do sistema são desenvolvidas em paralelo, e integradas quando completas. Não implica, requer ou pressupõe desenvolvimento iterativo ou em cascata ambos são estratégias de retrabalho. A alternativa ao desenvolvimento incremental é desenvolver todo o sistema com uma integração única.



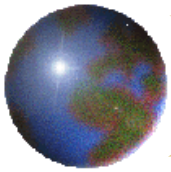
Modelo Incremental - Etapas

O projeto em si consiste da etapa de inicialização e iteração.



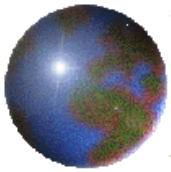
Modelo Incremental - Etapas

A **etapa de inicialização** cria uma versão base do sistema. O objetivo desta implementação inicial é criar um produto para que o usuário possa avaliar. Ele deve oferecer um exemplo dos aspectos chave do problema e prover uma solução que seja simples o bastante para que possa ser compreendida e implementada facilmente. Para guiar o processo iterativo, uma lista de controle de projeto é criada. Ela conterá um registro de todas as tarefas que necessitam ser realizadas. Isto inclui itens tais como novas características a serem implementadas e áreas para serem projeto na solução atual. A lista de controle deve ser continuamente revisada como um resultado da fase de análise.



Modelo Incremental - Etapas

A **etapa iterativa** envolve o re-projeto e implementação das tarefas da lista de controle do projeto e a análise da versão corrente do sistema. O objetivo para o projeto de implementação de qualquer iteração é ser simples, direto e modular, preparado para suportar re-projeto neste estágio ou como uma tarefa a ser adicionada na lista de controle do projeto. O código pode, em alguns casos, representar uma fonte maior da documentação do sistema. A análise de uma interação é baseada no feedback do usuário, e facilidades da análise do programa disponíveis. As estruturas de análise envolvidas são a modularidade, usabilidade, reusabilidade, eficiência e obtenção dos objetivos. A lista de controle do projeto é modificada à luz dos resultados da análise.



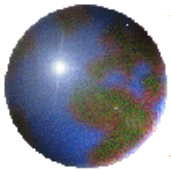
Modelo Incremental - Exemplos

- **1º Incremento: Funções de gerenciamento de arquivos, edição e produção de documentos;**
- **2º Incremento: Recursos mais sofisticados de edição e produção de documentos;**
- **3º Incremento: Revisão ortográfica e gramatical;**
- **4º Incremento: Recursos avançados de formatação de página.**

Modelo Incremental - Vantagens

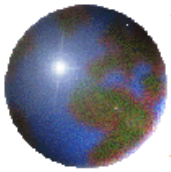
Modelo Incremental - Vantagens

- Riscos associados ao desenvolvimento de incrementos são menores, devido ao seu tamanho reduzido;
- Se um grande erro é cometido, apenas o último incremento é descartado;
- Reduzindo o tempo de desenvolvimento de um sistema, as chances de mudanças nos requisitos do usuário durante o desenvolvimento são menores;



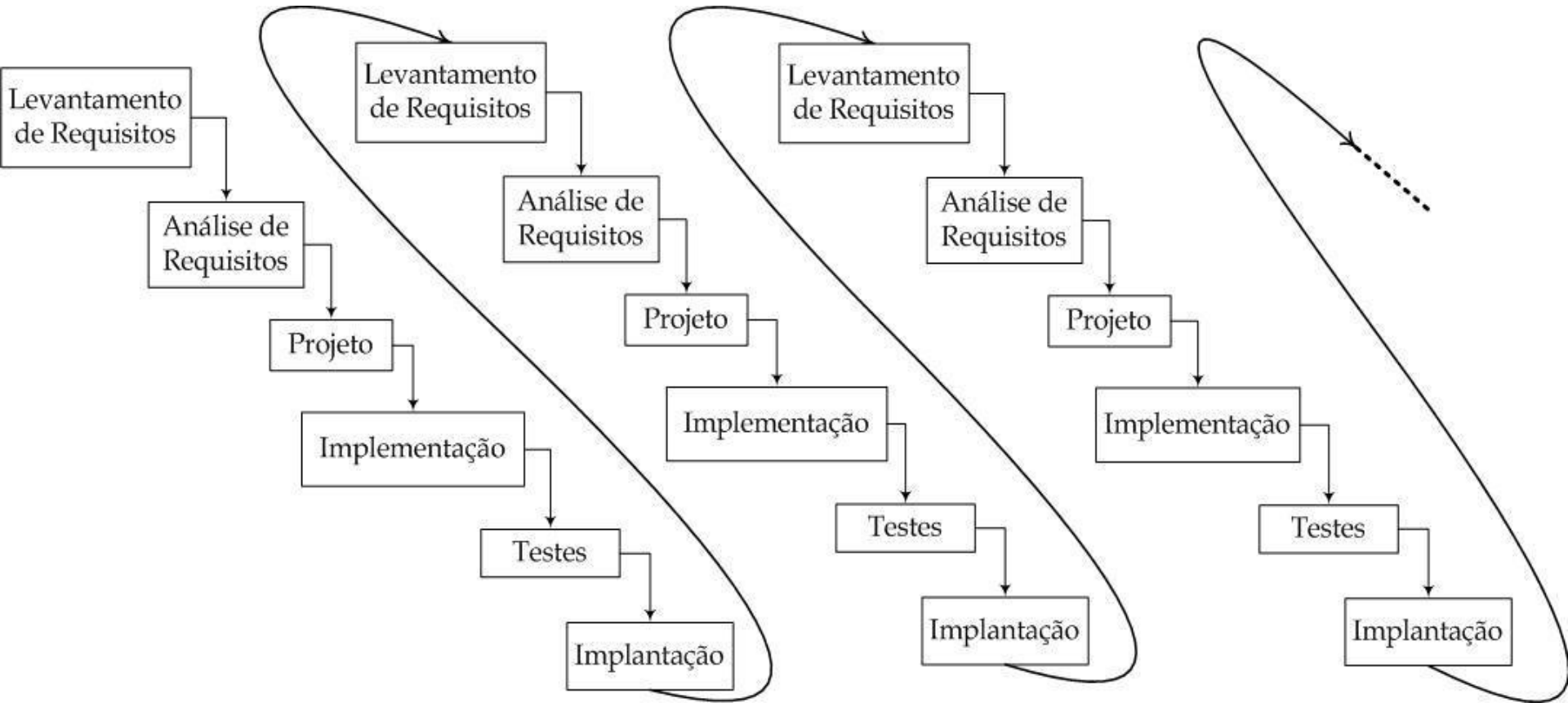
Modelo Incremental - Vantagens

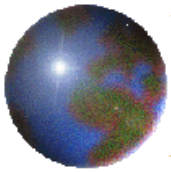
- Inclui o uso do software pelo usuário para que as mudanças sejam feitas de acordo com o mesmo;
- Melhor gerenciamento de riscos, porque você pode confirmar o resultado com o cliente depois de cada versão do sistema e sempre verificar se estão fazendo o que está de acordo com o plano ou não, e corrigi-los na próxima versão do software;
- Melhor custo e menos tempo são necessários para se entregar a primeira versão.



Modelo Incremental - Desvantagens

- **Problemas em montar um contrato para atender ambas as partes interessadas;**
- **Podem surgir problemas relativos à arquitetura do sistema, porque nem todos os requisitos estão reunidos na frente de todo o ciclo de vida do software;**
- **O modelo Incremental precisa ser relativamente pequeno;**


Levantamento de Requisitos
Análise de Requisitos
Projeto
Implementação
Testes
Implantação
Levantamento de Requisitos
Análise de Requisitos
Projeto
Implementação
Testes
Implantação
Levantamento de Requisitos
Análise de Requisitos
Projeto
Implementação
Testes
Implantação

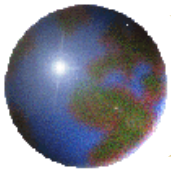


- **Número de iterações não pode ser definido no início do processo.**
- **O fim do processo não pode ser previamente definido.**
- **Gerenciamento e manutenção do sistema completo podem se tornar complexos.**
- **Gerenciamento do custo é mais complexo devido ao número de iterações (verba pode acabar).**

Modelo RAD

Rapid Application Development

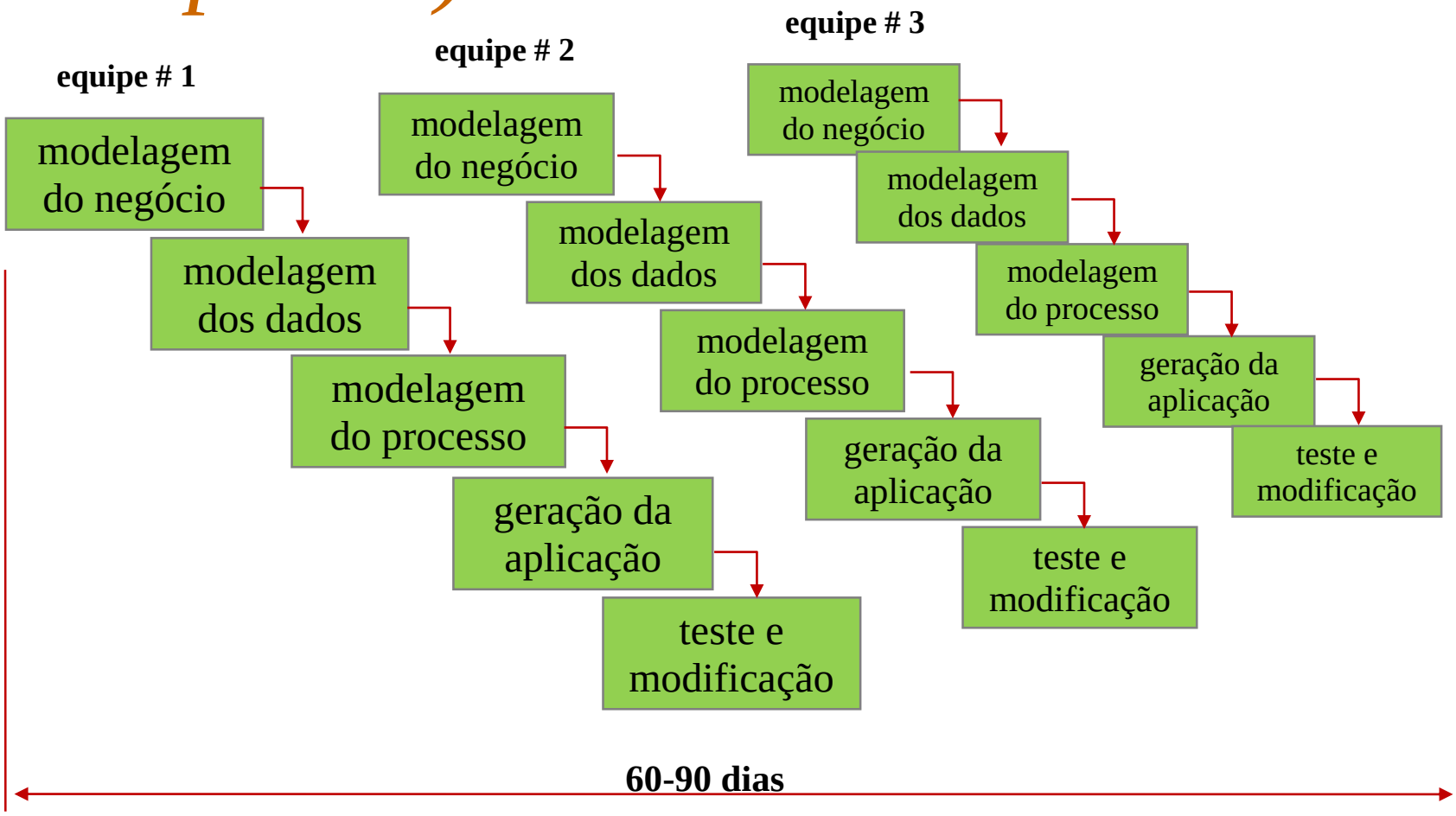
(Desenvolvimento rápido de aplicações)



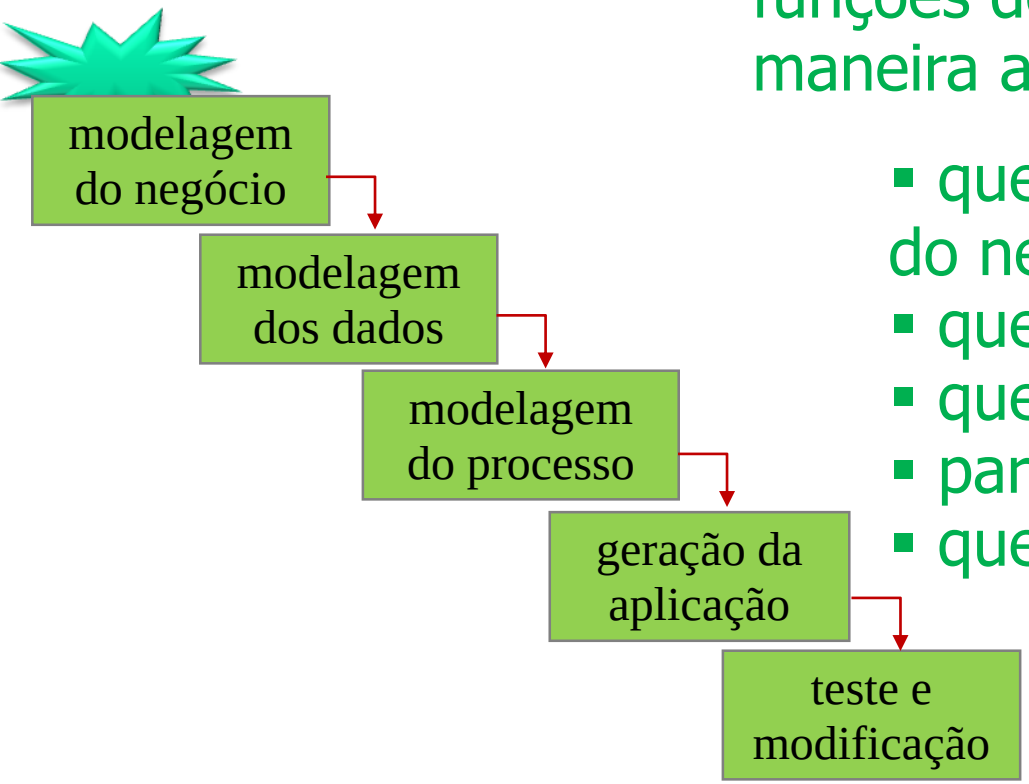
Modelo RAD (Rapid Application Development)

- **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 (Rapid Application Development)



Modelo RAD (Rapid Application Development)

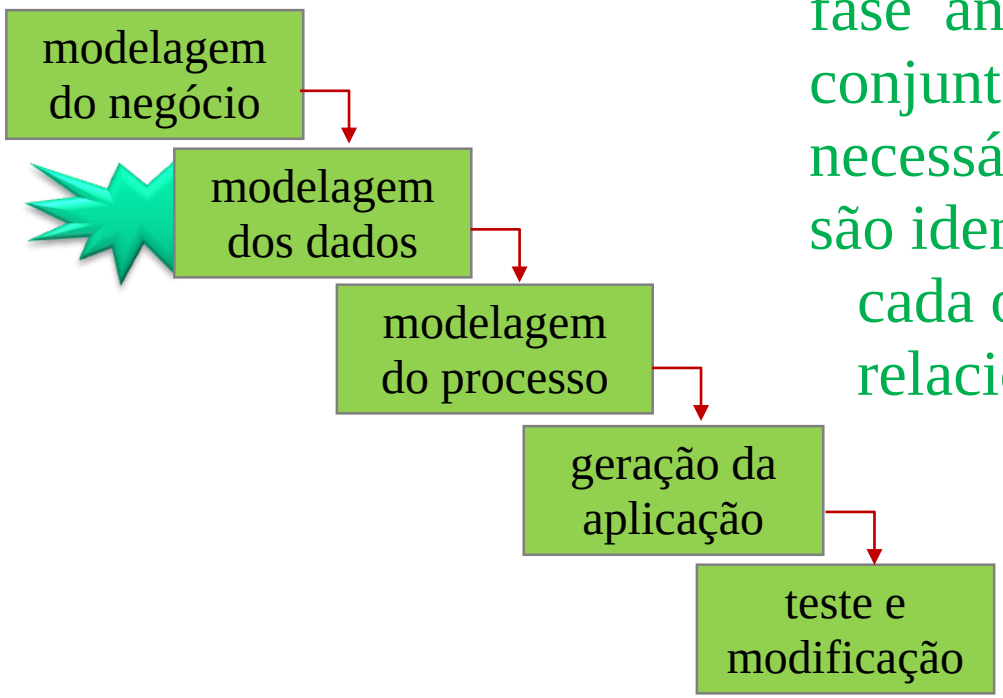


Modelagem do negócio:

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?

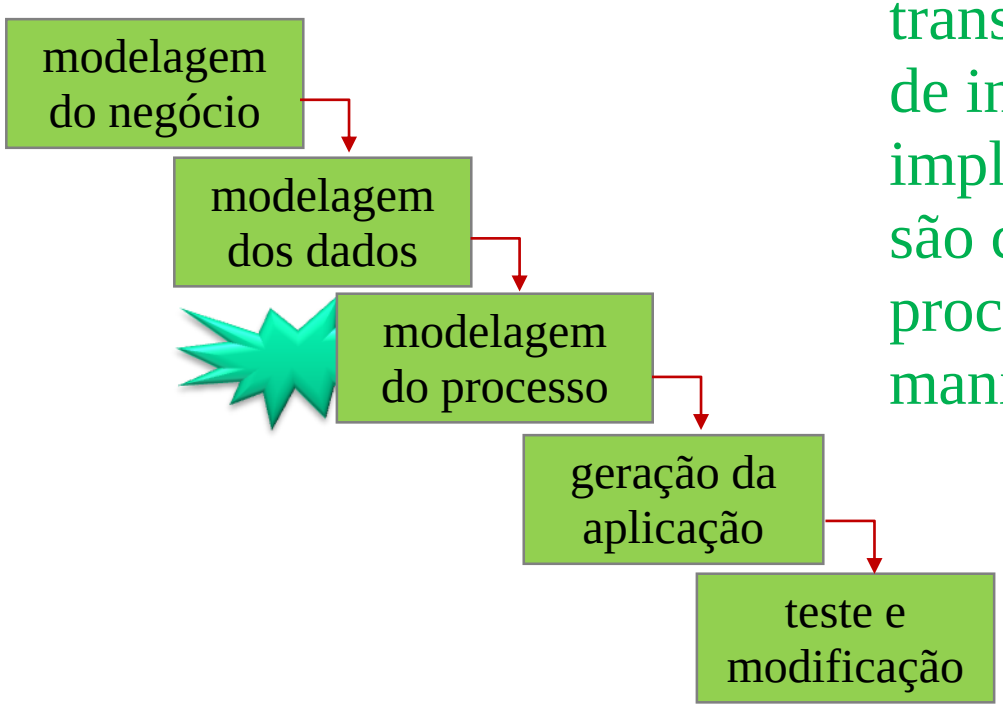
Modelo RAD (Rapid Application Development)



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

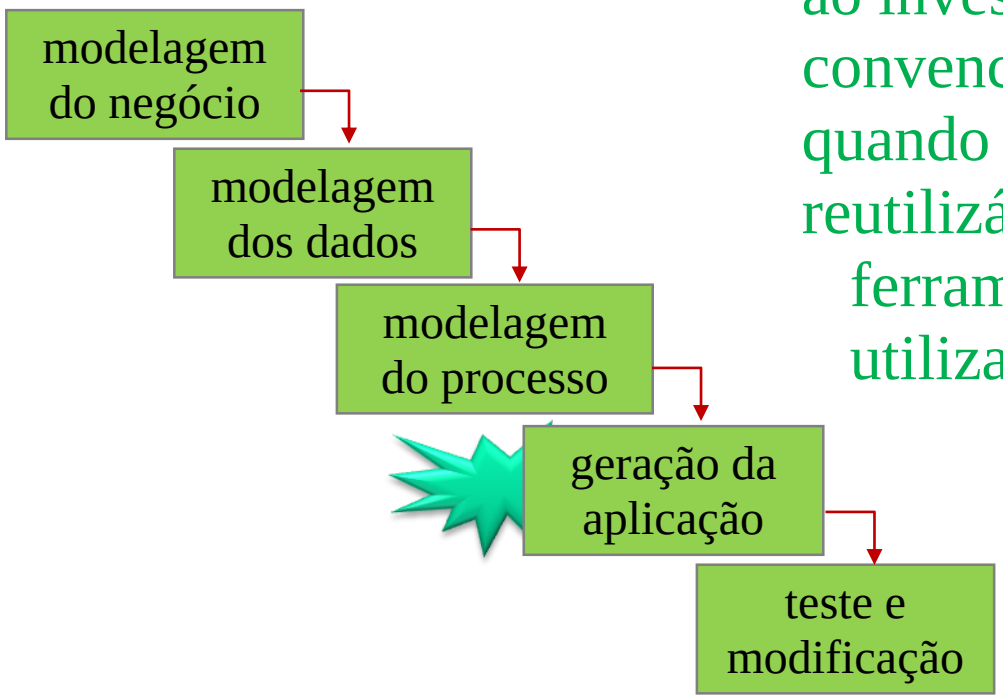
Modelo RAD (Rapid Application Development)



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

Modelo RAD (Rapid Application Development)



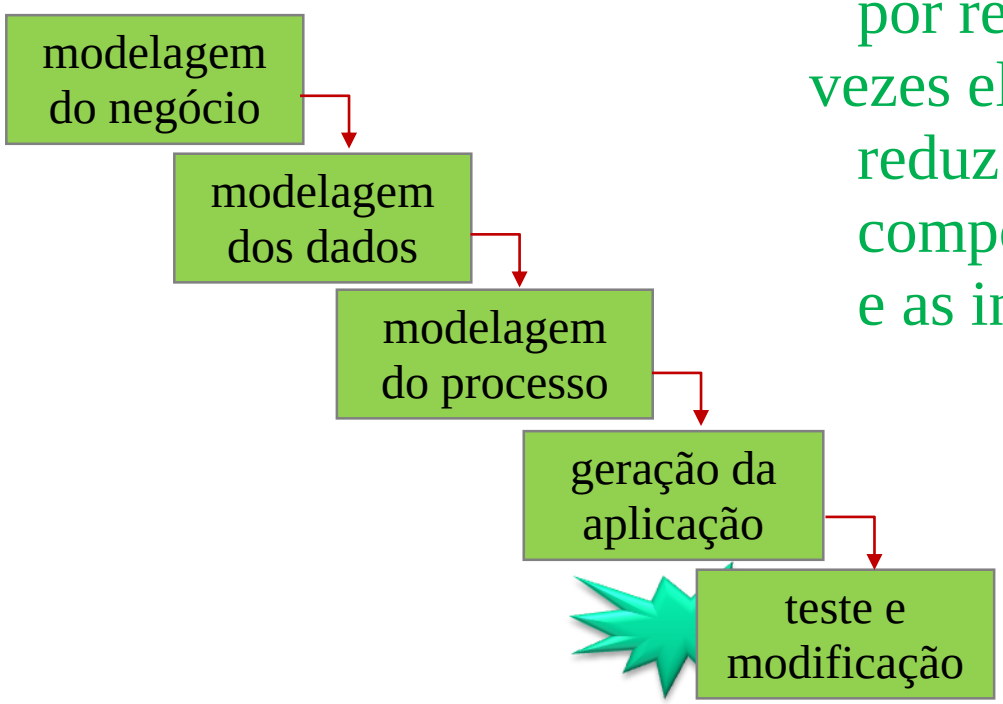
Geração da aplicação:

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;

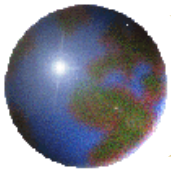
ferramentas automatizadas são utilizadas para gerar software

Modelo RAD (Rapid Application Development)



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

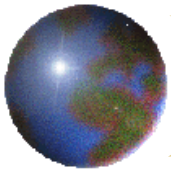


Modelo RAD (Rapid Application Development)

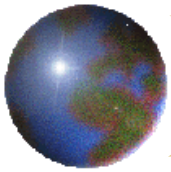
- **Quando usar?**
  - as restrições de tempo impostas pelo projeto demandam um escopo de escala
  - quando a aplicação pode ser modularizada de forma que cada grande função possa ser completada em menos de 3 meses
  - cada grande função pode ser alocada para uma equipe distinta e, depois são integradas para formar o todo

Modelo RAD (Rapid Application Development)

- **Problemas com modelo**
  - 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



Visão Genérica



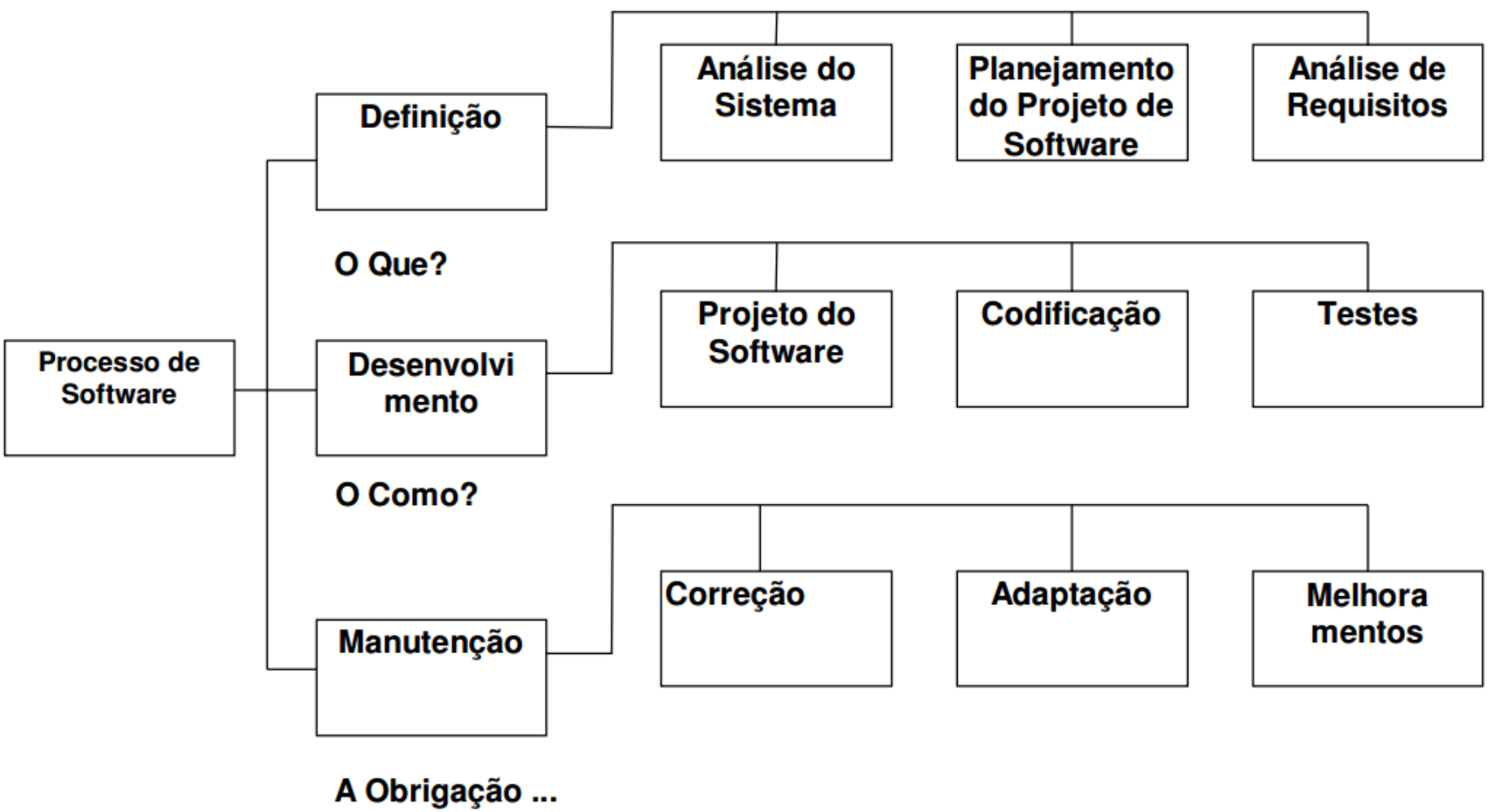
Engenharia de Software
uma visão genérica

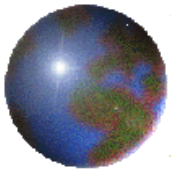
O processo de desenvolvimento de software contém 3 fases genéricas, independentes do modelo de engenharia de software escolhido:

- ☆ DEFINIÇÃO
- ☆ DESENVOLVIMENTO
- ☆ MANUTENÇÃO

Engenharia de Software

uma visão genérica

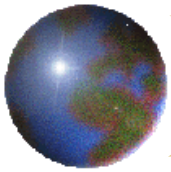




Engenharia de Software
uma visão genérica

FASE DE DEFINIÇÃO: *"o quê"* será desenvolvido

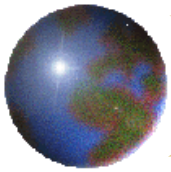
- ***Análise do Sistema*: define o papel de cada elemento num sistema baseado em computador, atribuindo em última análise, o papel que o software desempenhará.**
- ***Planejamento do Projeto de Software*: assim que o escopo do software é estabelecido, os riscos são analisados, os recursos são alocados, os custos são estimados, e tarefas e programação de trabalho são definidas.**
- ***Análise de Requisitos*: o escopo definido para o software proporciona uma direção, mas uma definição detalhada do domínio da informação e da função do software é necessária antes que o trabalho inicie.**



Engenharia de Software
uma visão genérica

FASE DE DESENVOLVIMENTO: "como" o software vai ser desenvolvido

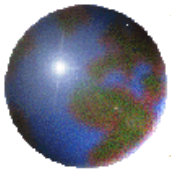
- ***Projeto de Software*: traduz os requisitos do software num conjunto de representações (algumas gráficas, outras tabulares ou baseadas em linguagem) que descrevem a estrutura de dados, a arquitetura do software, os procedimentos algorítmicos e as características de interface**
- ***Codificação*: as representações do projeto devem ser convertidas numa linguagem artificial (a linguagem pode ser uma linguagem de programação convencional ou uma linguagem não procedimental) que resulte em instruções que possam ser executadas pelo computador**
- ***Realização de Testes do Software*: logo que o software é implementado numa forma executável por máquina, ele deve ser testado para que se possa descobrir defeitos de função, lógica e implementação**



Engenharia de Software
uma visão genérica

FASE DE MANUTENÇÃO: concentra-se nas "mudanças" que ocorrerão depois que o software for liberado para uso operacional

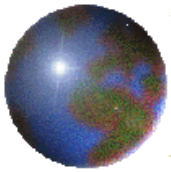
- *Correção*
- *Adaptação*
- *Melhoramento Funcional*



Engenharia de Software
uma visão genérica

Correção: **mesmo com as melhores atividades de garantia de qualidade de software, é provável que o cliente descubra defeitos no software. A manutenção *corretiva* muda o software para corrigir defeitos.**

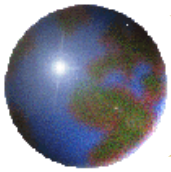
***Adaptação*: com o passar do tempo, o ambiente original (por exemplo a CPU, o sistema operacional e periféricos) para o qual o software foi desenvolvido provavelmente mudará. A manutenção *adaptativa* muda o software para acomodar mudanças em seu ambiente.**



Engenharia de Software
uma visão genérica

Melhoramento Funcional: a medida que o software é usado, o cliente/usuário reconhecerá funções adicionais que oferecerão benefícios.

A *manutenção perfectiva* estende o software para além de suas exigências funcionais originais.



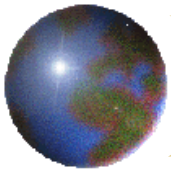
Engenharia de Software
uma visão genérica

ATIVIDADES DE PROTEÇÃO as fases e etapas correlatas descritas são complementadas por uma série de atividades de proteção.

Revisões: efetuadas para garantir que a qualidade seja mantida à medida que cada etapa é concluída.

Documentação: é desenvolvida e controlada para garantir que informações completas sobre o software estejam disponíveis para uso posterior.

Controle das Mudanças: é instituído de forma que as mudanças possam ser aprovadas e acompanhadas.



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 1987)