Pós-Graduação em Engenharia de Software

Disciplina
Processo de Software
24h
04.11 a 09.12

Segundas-feiras

Revisão

Ciclo de Vida de Software

software development life-cycle (SDLC)

Representação abstrata, esquemática e simplificada do processo de desenvolvimento do Software, desde a sua concepção até a sua descontinuidade ou substituição.

Mostrando a sequência dos passos e os produtos e sub-produtos gerados ao longo do processo.

Ciclo de Vida de um Software

Manutenção Evolutiva

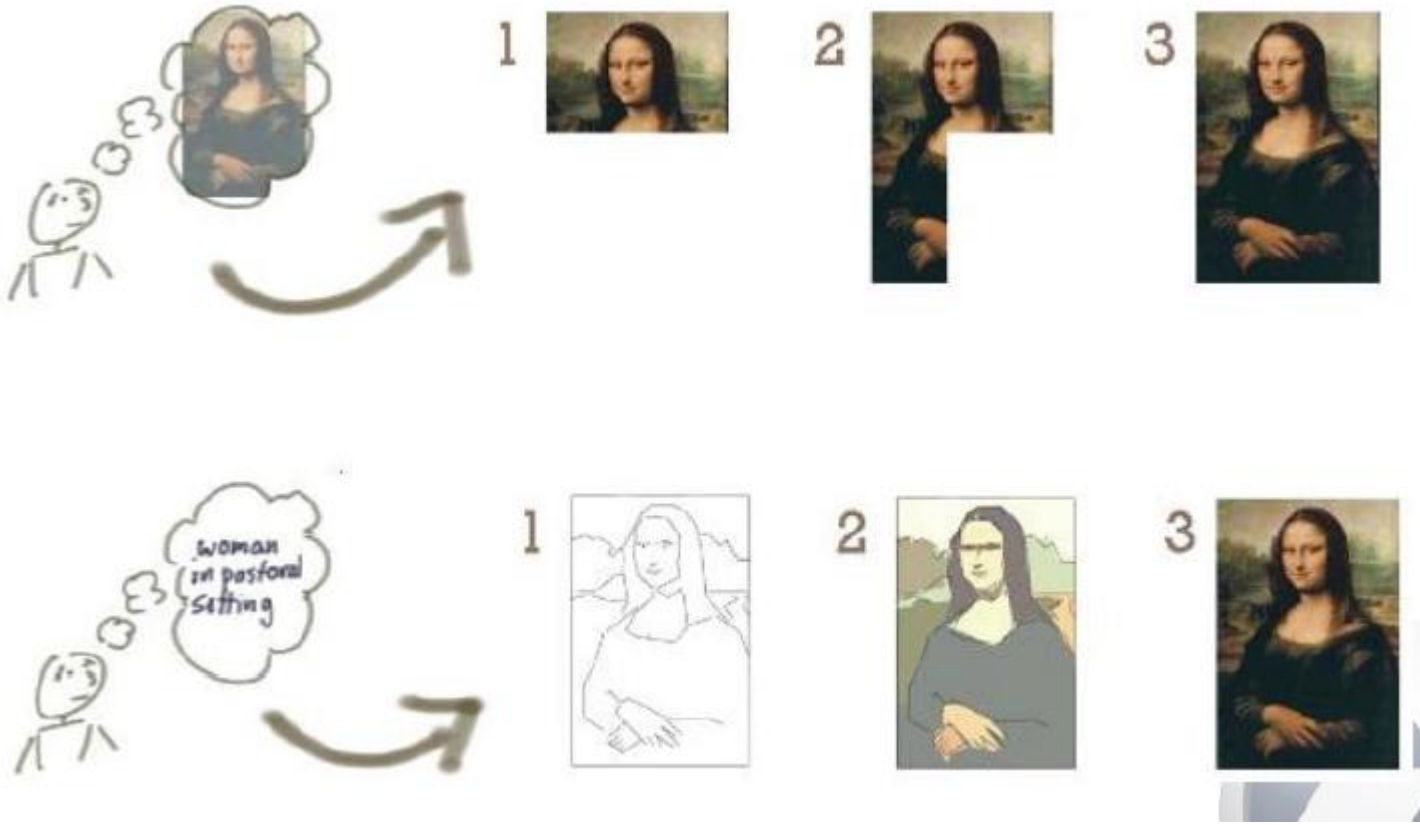
Envolve todas as mudanças, inclusões, exclusões, alterações e aprimoramentos efetuados em um sistema com o intuito de satisfazer as novas funcionalidades, ampliações e modificações solicitadas pelo usuário.

Manutenção Corretiva

É diagnosticar e corrigir os problemas do sistema como, por exemplo, falhas de funcionalidades, erros software (código fonte), entre outros.

Revisão

Iterativo x Incremental



O PROCESSO ITERATIVO

- ✓ Ideia de " melhorar (ou refinar) pouco - a - pouco " o sistema (iterações);
- ✓ A cada iteração identifica-se e especifica-se os requisitos relevantes, cria um projeto utilizando a arquitetura escolhida como guia, implementa o projeto em componentes e verifica se esses componentes satisfazem os requisitos.
- ✓ Se a iteração atinge os seus objetivos, o desenvolvimento prossegue com a próxima iteração

O PROCESSO INCREMENTAL

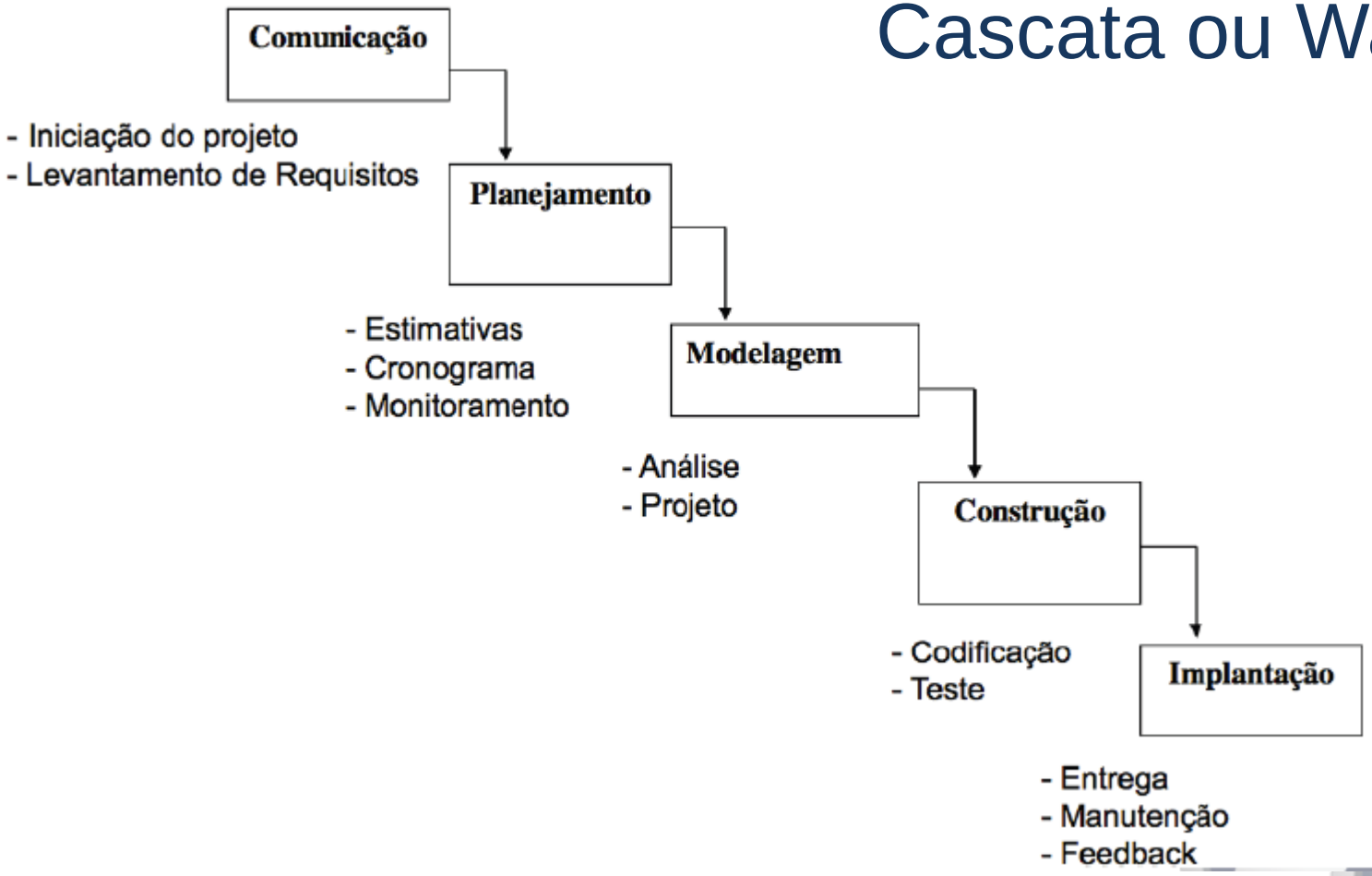
- ✓ Ideia de " aumentar (alargar) pouco-a-pouco " o âmbito do sistema.
- ✓ Um incremento não é necessariamente a adição do código executável correspondente aos casos de uso que pertencem à iteração em andamento. Especialmente nas primeiras fases do ciclo de desenvolvimento, os desenvolvedores podem substituir um projeto superficial por um mais detalhado ou sofisticado. Em fases avançadas os incrementos são tipicamente aditivos.

Modelos de Processo

- ✓ Codifica-Remenda;
- ✓ Clássico ou CASCATA / Waterfall;
- ✓ V-Model
- ✓ Sashimi;
- ✓ Espiral;
- ✓ Prototipagem Evolutiva;
- ✓ Entrega por Estágios;
- ✓ Entrega Evolutiva
- ✓ RAD (Rapid Appilcation Development)
- ✓ Orientado a Reuso
- ✓ RUP* (Rational Unified Process)

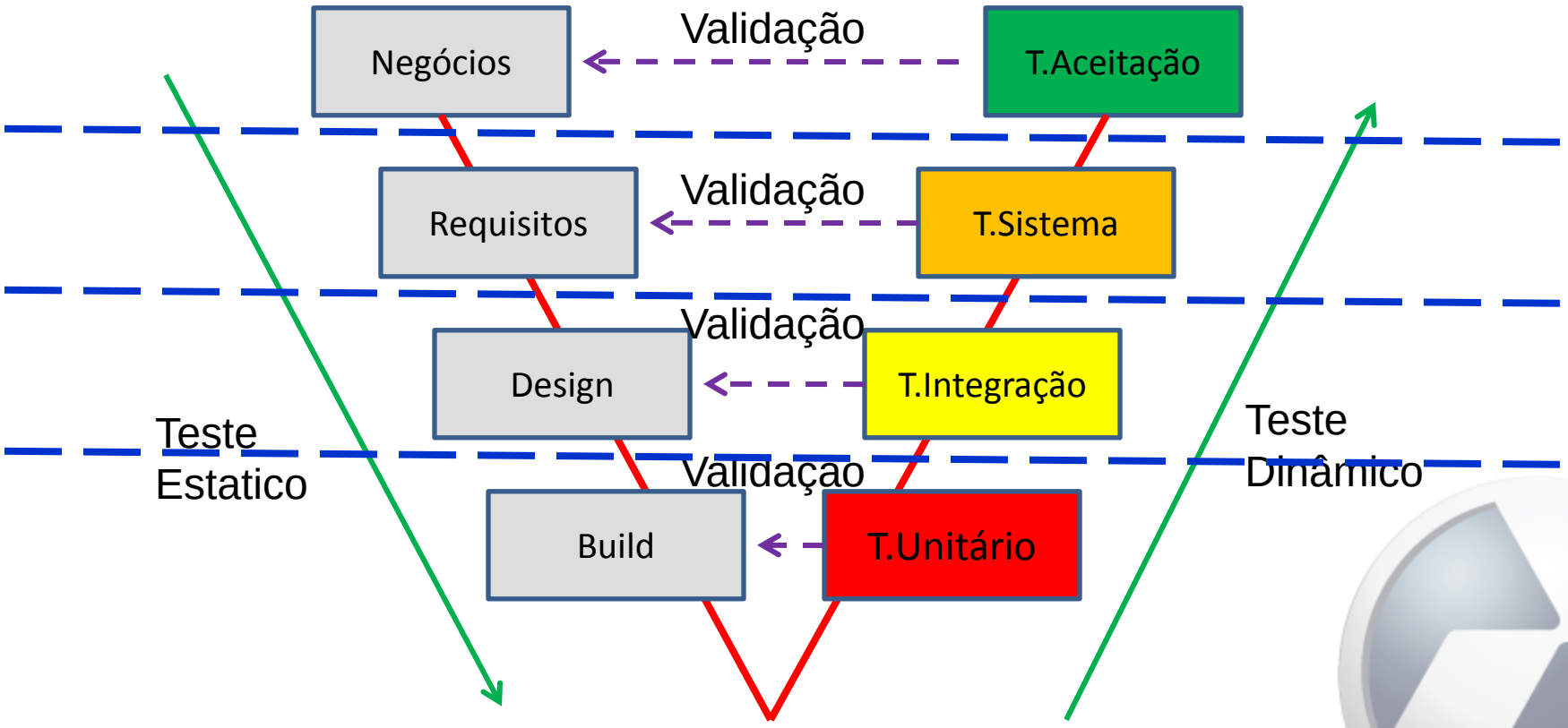
Revisão

Cascata ou Waterfall



Pressman

V-Model


Negócios
Validação
T.Aceitação
Requisitos
Validação
T.Sistema
Design
Validação
T.Integração
Build
Validação
T.Unitário
Teste
Estatico
Teste
Dinâmico

Revisão

"Sashimi"

Origem:
Variação do Cascata/ Waterfall

Similar to the popular Japanese Sashimi food, the Sashimi Waterfall model phases overlap in an iterative fashion.

Questions

Faculdade IMPACTA TECNOLOGIA

Agenda Aula-3

Processo de Software

- ✓ História
- ✓ Conceitos da Engenharia de Software
- ✓ Processo de Software
- ➢ Modelos de Processo
- ✓ Introdução Métodos Ageis
- ✓ Introdução Scrum
- ✓ Introdução UP
- ✓ Introdução RUP

Modelos de Processo

- ✓ Codifica-Remenda;
- ✓ Clássico ou CASCATA / Waterfall;
- ✓ V-Model
- ✓ Sashimi;
- ✓ Espiral;
- ✓ Prototipagem Evolutiva;
- ✓ Entrega por Estágios;
- ✓ Entrega Evolutiva
- ✓ RAD (Rapid Appilcation Development)
- ✓ Orientado a Reuso
- ✓ RUP* (Rational Unified Process)

Origem:
Cascata + Prototipação + análise de riscos

Espiral

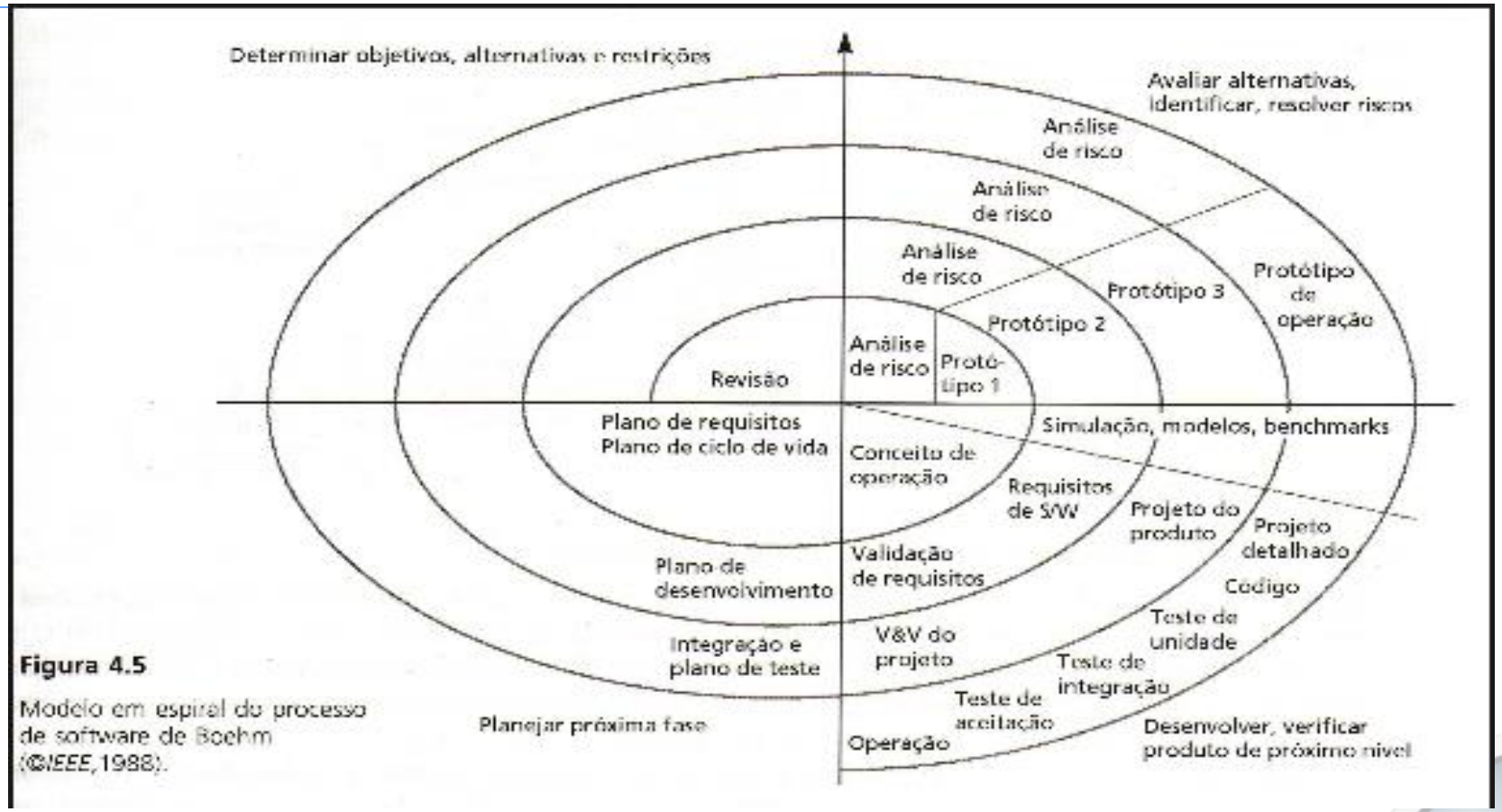


Figura 4.5

Modelo em espiral do processo de software de Boehm (©IEEE, 1988).

O modelo Espiral, proposto por Barry Boehm

Faculdade IMPACTA TECNOLOGIA

Espiral

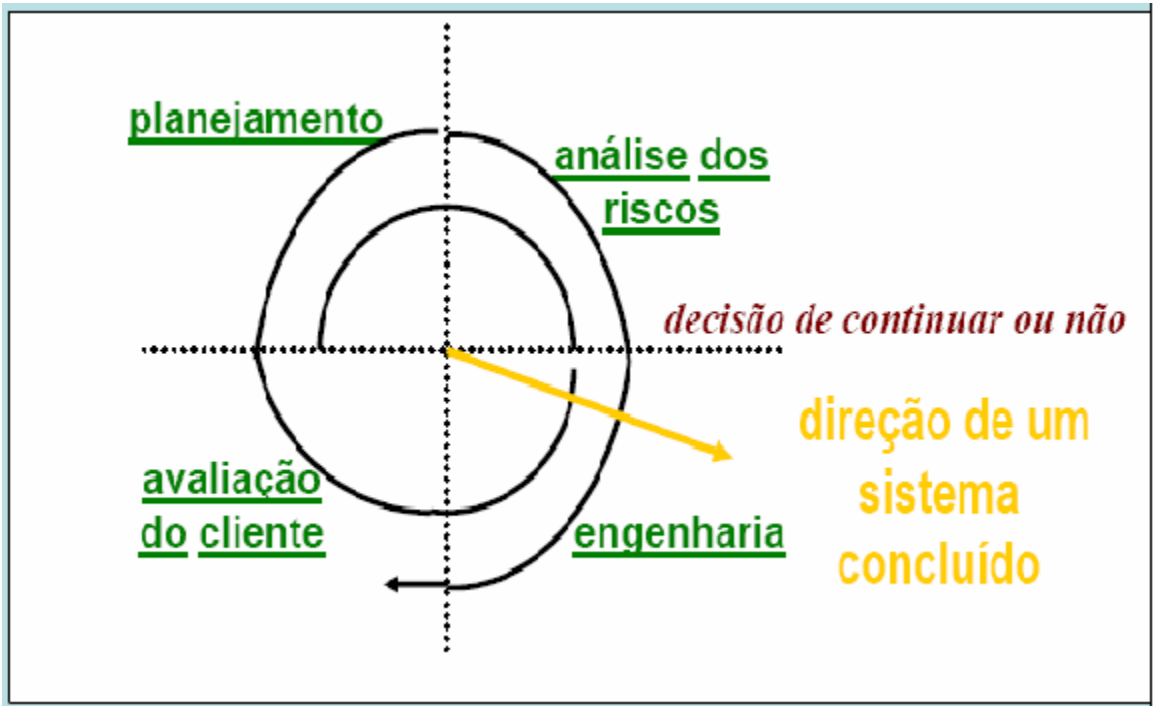
- ✓ É desenvolvido em uma série de iterações;
- ✓ Cada nova iteração corresponde a uma volta na espiral;
- ✓ Cada volta na espiral (iniciando a partir do centro e avançando para fora) representa uma nova fase do processo.

Espiral

- ✓ Esse processo evolutivo permite que novas versões possam ser construídas progressivamente.
- ✓ Tipicamente, o modelo pode ser dividido em 3 ou 6 regiões.
- ✓ Produtos gerados em prazos curtos;
- ✓ Características e recursos são agregados na medida em que a experiência descobre sua necessidade

Faculdade IMPACTA TECNOLOGIA

Espiral



Visão Geral

Planejamento:

São definidos os objetivos, as restrições para o projeto e é preparado um plano de gerenciamento detalhado. São identificados os riscos do projeto e, dependendo dos riscos, poderão ser planejadas estratégias alternativas.

Analise dos Riscos

Para cada um dos riscos de projeto identificados, é realizada uma análise detalhada e são tomadas providências para reduzir esses riscos.

Engenharia

Depois da avaliação dos riscos, é escolhido um modelo de desenvolvimento para o sistema.
Por exemplo, o modelo cascata.

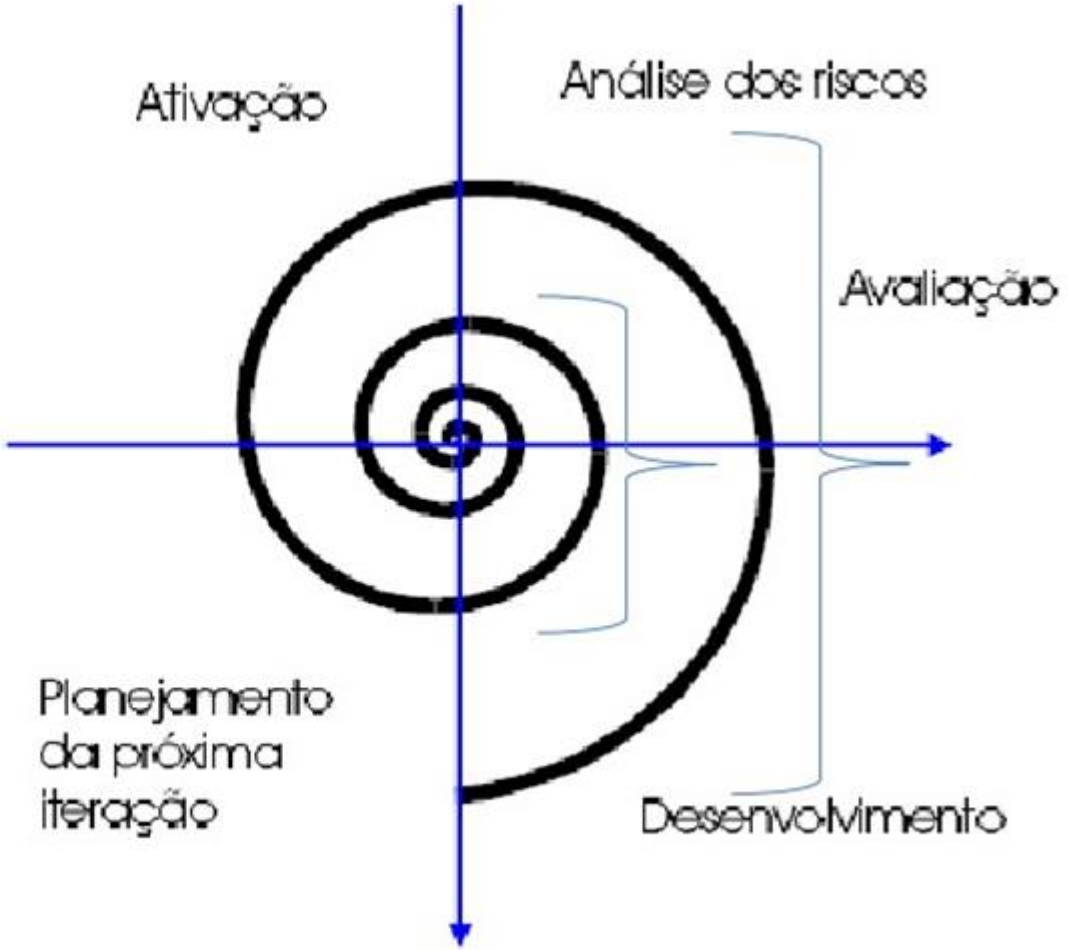
Avaliação do Cliente

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

Espiral

A cada iteração novos recursos podem ser adicionados visando correção

A manutenção é usada para identificar problemas



Pontos Fortes

- ✓ Modelo evolutivo possibilita uma maior integração entre as fases e facilita a depuração e a manutenção do Sistema.
- ✓ execução de atividades de verificação presentes ao final de cada iteração que permitem um melhor controle gerencial sobre o projeto
- ✓ Permite que o projetista e cliente possam entender e reagir aos riscos em cada etapa evolutiva

Pontos Fracos

- ✓ Avaliação dos riscos exige muita experiência.
- ✓ Requer uma gestão muito sofisticada para ser previsível e confiável

Prototipagem

Prototipagem

Protótipos são, normalmente, desenvolvidos de forma rápida e representam uma versão simplificada do software, que implementam certas caracteristicas e funcionalidade.

O protótipo é um modelo operacional do software a ser desenvolvido

(Budde e Zullighoven, 1990).

Prototipagem

Auxiliam a especificação de requisitos, e servem de base para subsidiar tomadas de decisão e como forma de ganhar experiência prática.

(Budde e Zullighoven, 1990).

Prototipagem

Classificam-se pelas técnicas de construção:

- ✓ Horizontal: implementação superficial de apenas uma camada do software, geralmente a camada de interface com o usuário; e

- ✓ Vertical: uma parte específica do software é implementada por completo.

(Lichter *et al.* (1993)

Prototipagem

Protótipos podem ter:

- ✓ Alta ou
- ✓ Baixa fidelidade

Metodologia de Prototipagem

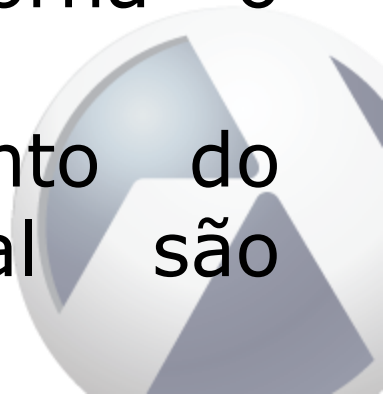
- ✓ Descartáveis ou
- ✓ Evolutivas ou evolucionárias

Prototipagem Descartável

- ✓ Protótipo (*throw-away*), após sua utilização, são deixados de lado, servindo apenas para consultas.
- ✓ O processo de desenvolvimento do software real é totalmente independente do processo utilizado para a elaboração do protótipo.
- ✓ Normalmente possuem baixa fidelidade, pois são criados para auxiliar a especificação de requisitos e não deveriam integrar o desenho do software.

Prototipagem Evolutiva

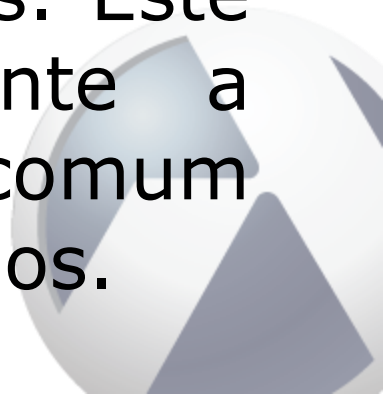
- ✓ Protótipos evolutivos (*evolutionary*) são criados nas fases iniciais do projeto e refinados ao longo do decorrer do processo de desenvolvimento do software.
- ✓ São interpretados como liberações. Incrementos de funcionalidade são incorporados ao protótipo, com fidelidade gradualmente aumentada, se torna o software final.
- ✓ Os processos de desenvolvimento do protótipo e do software real são essencialmente o mesmo.



No processo de prototipagem:

- ✓ A comunicação deve priorizar a interação e garantir a cooperação entre desenvolvedores e usuários.

- ✓ A comunicação deve ser gerenciada de forma centralizada e com ênfase no retorno das requisições dos usuários. Este requisito se aplica primordialmente a sistemas baseados na Web, onde é comum a existência de usuários desconhecidos.



Prototipagem

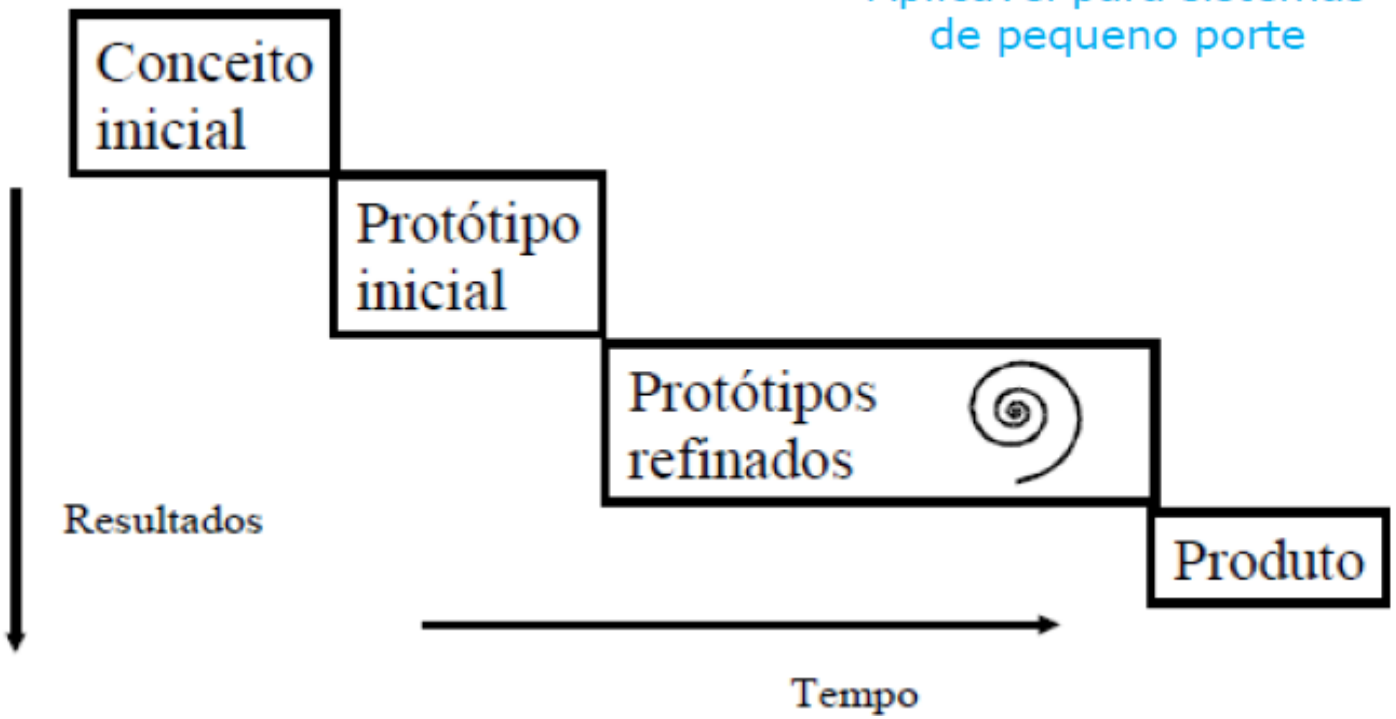
O processo de prototipagem:

- ✓ Acesso dos usuários finais ao protótipo deve ser limitado e gerenciado.
- ✓ No caso da prototipagem evolutiva, a comunicação deve ser assegurada e monitorada em todo o processo de desenvolvimento, e não apenas na especificação de requisitos e na implantação, quando ela é indispensável.

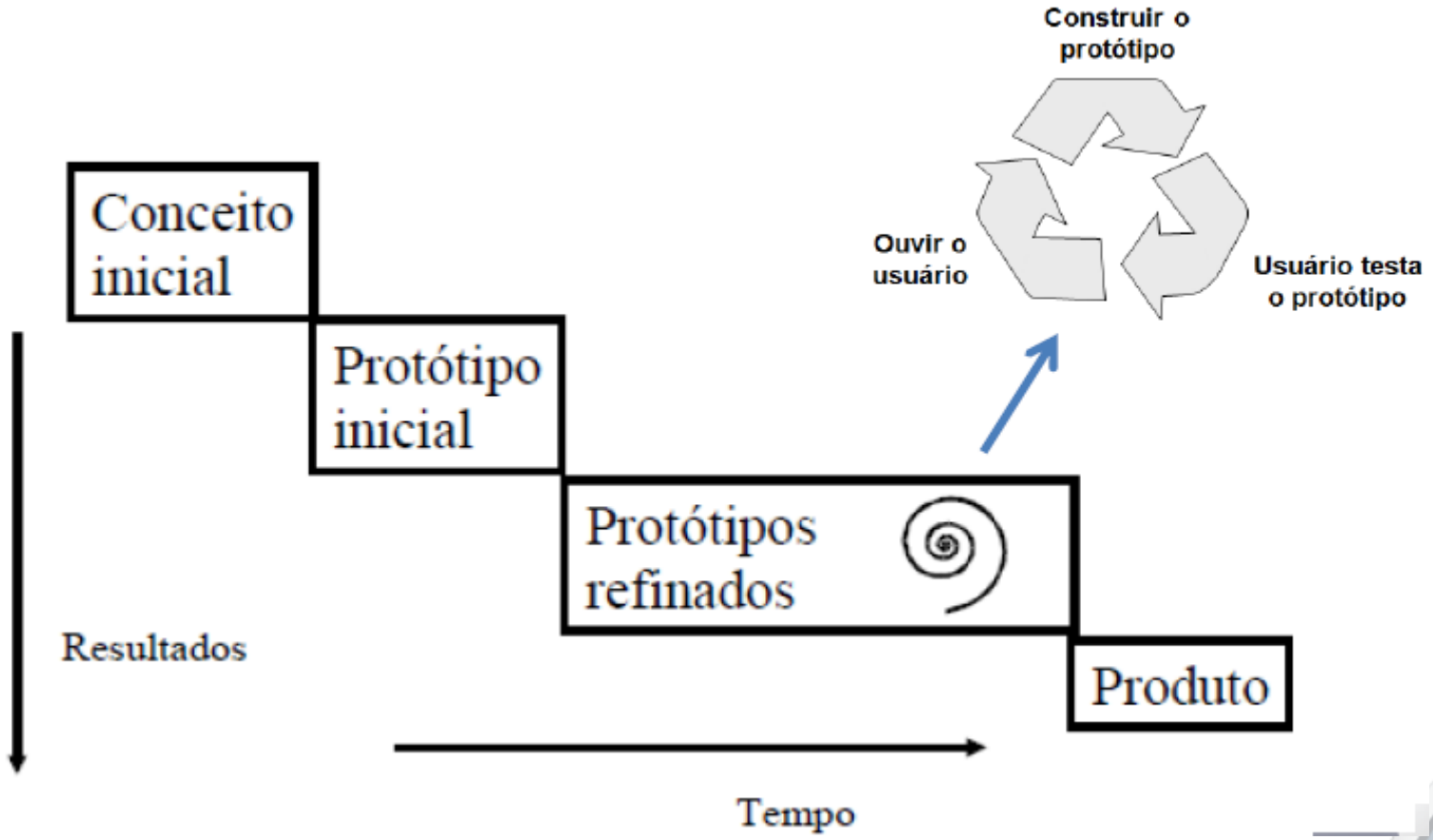
Paula Filho, W. P. (2001), Engenharia de Software, LTC, 2ª edição

Prototipagem Evolutiva

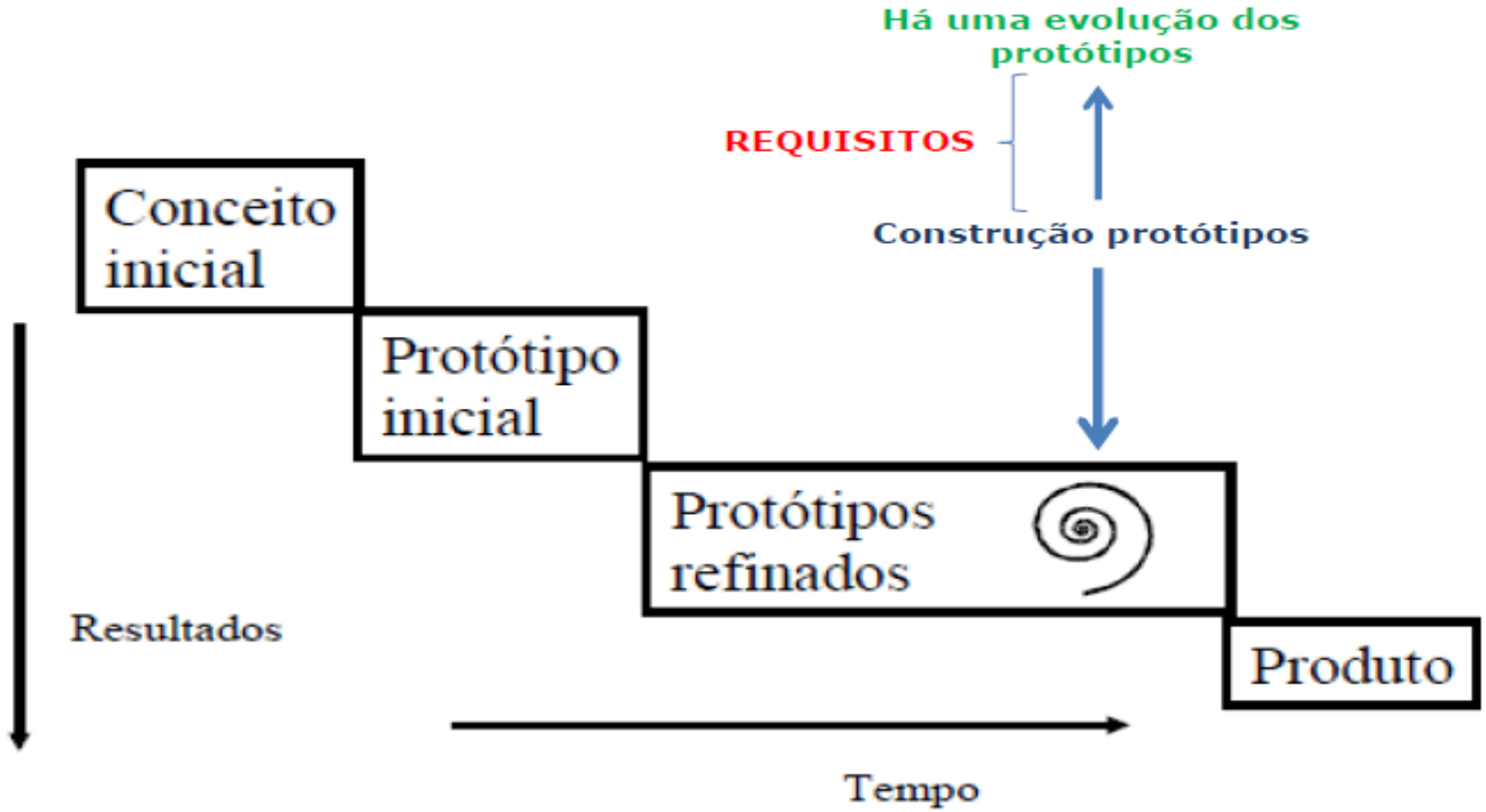
Aplicável para sistemas de pequeno porte



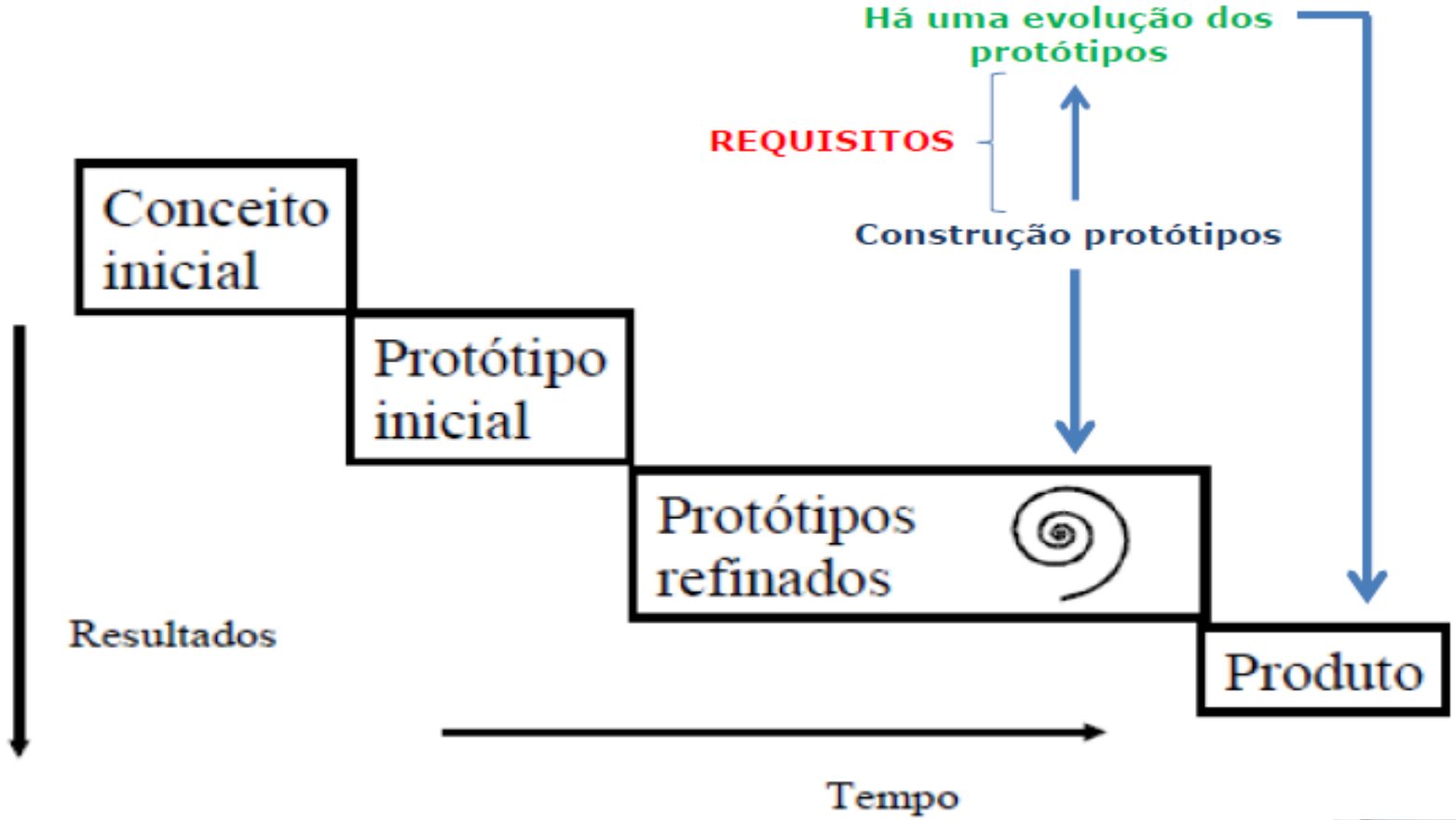
Prototipagem Evolutiva



Prototipagem Evolutiva



Prototipagem Evolutiva



Pontos Fortes

- ✓ Permite uma progressão de requisitos com o cliente;
- ✓ Possuem alta flexibilidade e visibilidade para os clientes

Pontos Fracos

- ✓ Fácil de esquecer a documentação;
- ✓ Análise e Projeto devem ser de excelente qualidade, para que não afetem os protótipos;
- ✓ Requer uma gestão muito sofisticada para ser previsível e confiável;
- ✓ Perda na qualidade: “ah, tá bom assim”, “não tem + tempo”

Questions


break

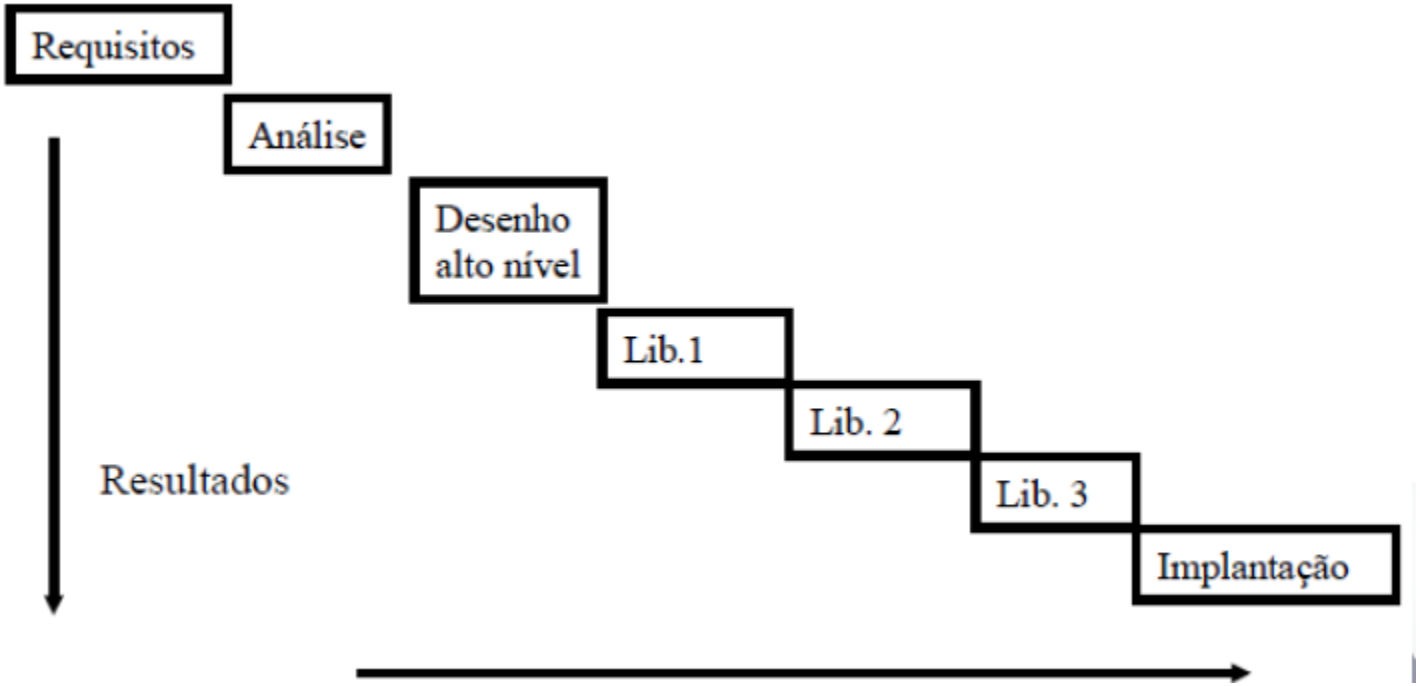
Entrega por Estágios

Origem: Cascata

Entrega por Estágios

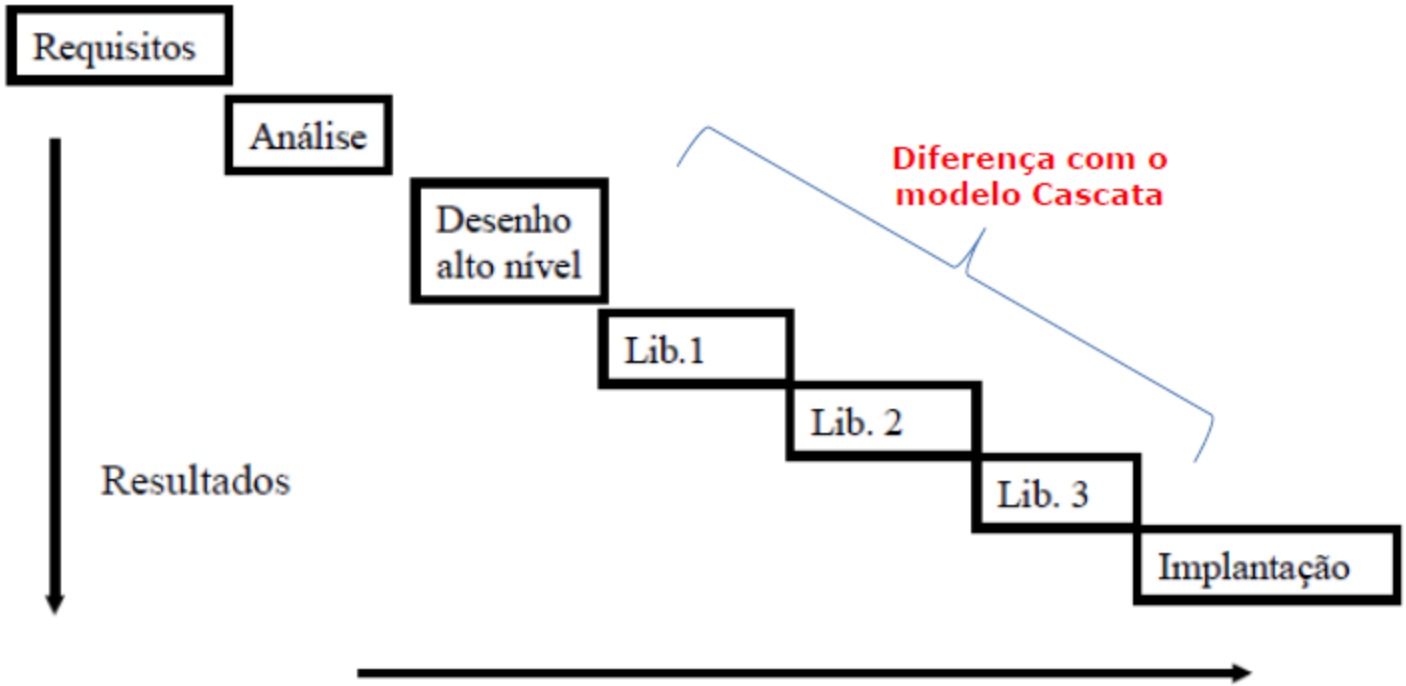
- ✓ Difere do modelo de ciclo de vida em cascata pela entrega ao cliente de liberações parciais do produto;
- ✓ Possui uma maior visibilidade do projeto, melhorando o relacionamento com o cliente.

Entrega por Estágios



Faculdade IMPACTA TECNOLOGIA

Entrega por Estágios



Entrega por Estágios

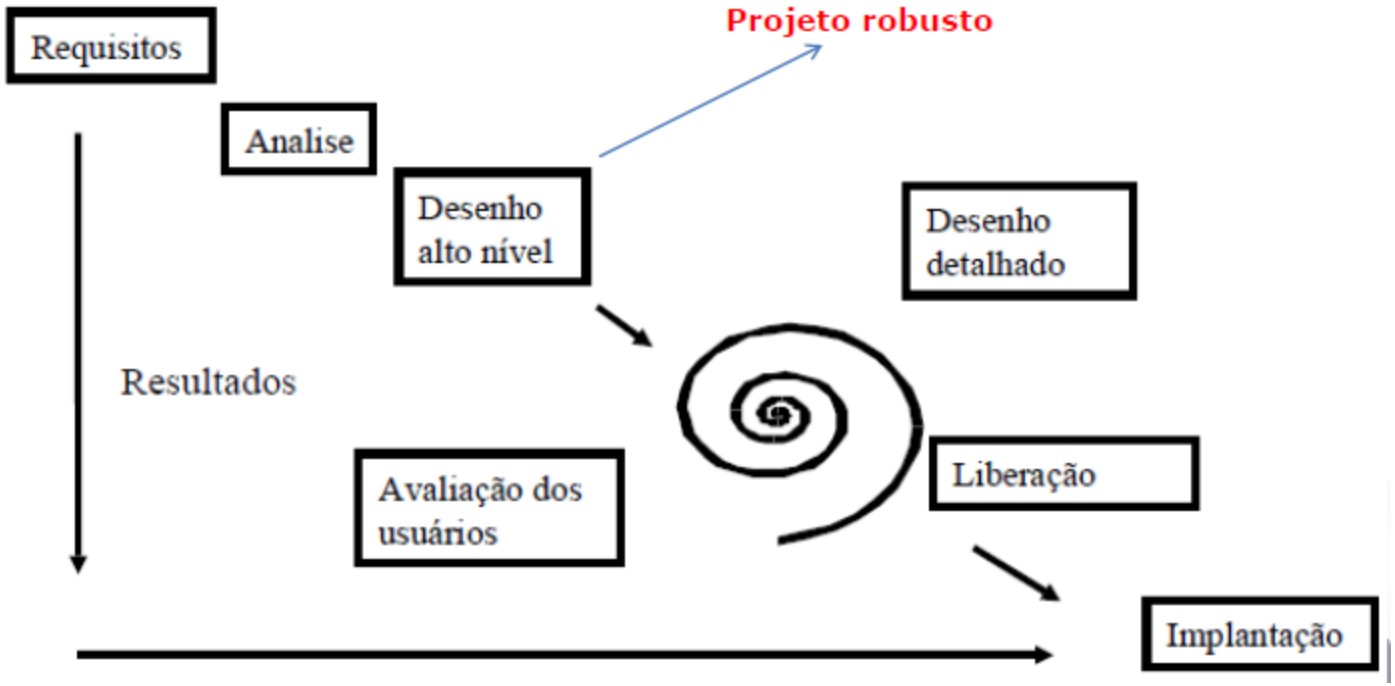
- Explicação:

Entrega Evolutiva

Origem

Cascata + Prototipagem Evolutiva

Entrega Evolutiva



Pontos Fortes

- ✓ Permite que em pontos bem definidos, os usuários possam avaliar partes do sistema
- ✓ Facilita o acompanhamento (gerentes e clientes) do progresso de cada projeto

Pontos Fracos

- ✓ Devem produzir uma arquitetura de produto robusta, que se mantenha íntegra ao longo dos ciclos de liberações parciais

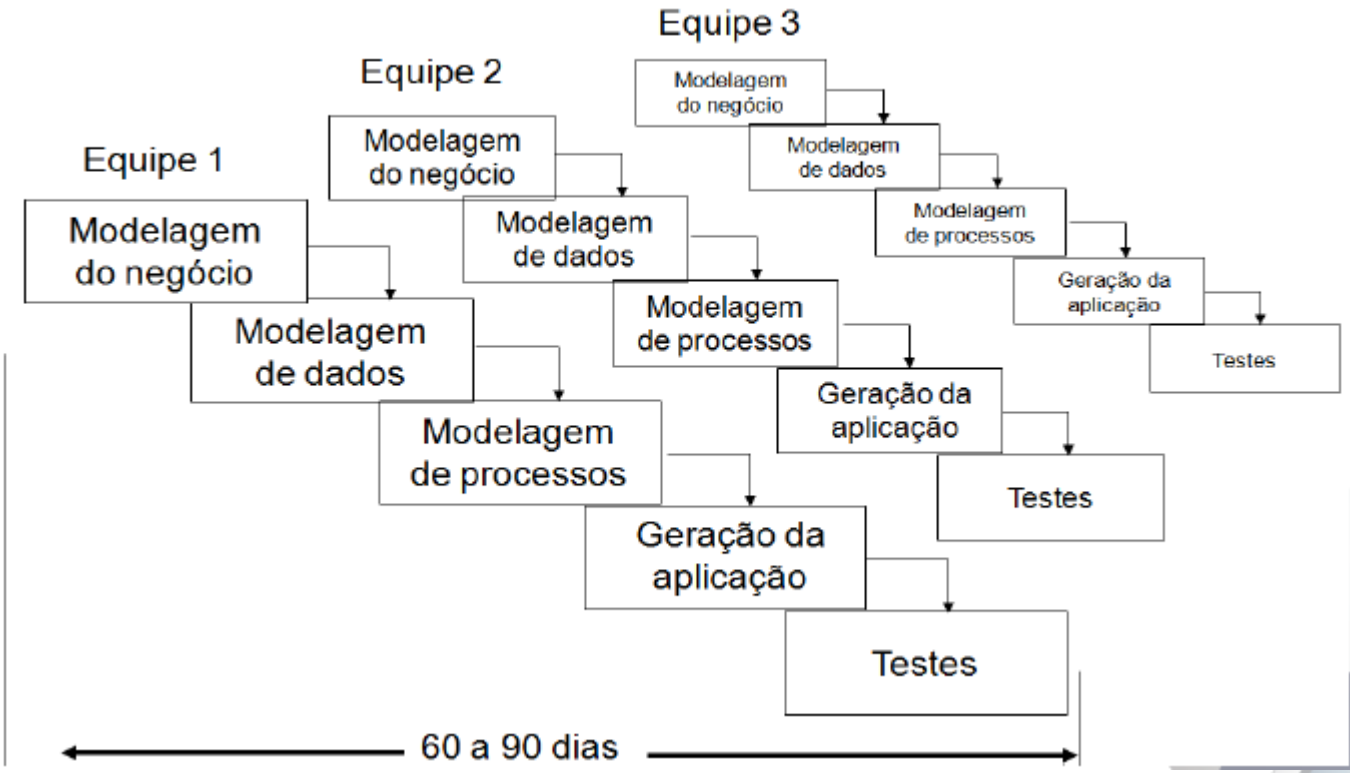
RAD
Rapid Application Development

RAD

Rapid Application Development (RAD) ou Desenvolvimento Rápido de Aplicação.

Modelo de processo de desenvolvimento de software iterativo e incremental que enfatiza um ciclo de desenvolvimento extremamente curto (entre 60 e 90 dias).

RAD


Equipe 3
Equipe 2
Equipe 1
Modelagem do negócio
Modelagem de dados
Modelagem de processos
Geração da aplicação
Testes
Modelagem do negócio
Modelagem de dados
Modelagem de processos
Geração da aplicação
Testes
Modelagem do negócio
Modelagem de dados
Modelagem de processos
Geração da aplicação
Testes
60 a 90 dias

O processo se divide em 5 fases:

1-Modelagem de Negócio (Na modelagem de negócio são levantados os processos suportados pelo sistema)

O fluxo de informações entre as funções de negócio é modelado:

- ✓ que informação direciona o processo de negócio?
- ✓ que informação é gerada?
- ✓ quem a gera?
- ✓ para onde vai à informação?
- ✓ quem a processa?

2-Modelagem dos dados

Extração dos principais objetos de dados a serem processados pelo sistema, baseados no Fluxo de Informações:

- ✓ qual a composição de cada um dos objetos de dados,
- ✓ onde costumam ficar,
- ✓ qual a relação entre eles
- ✓ quais as relações entre os objetos
- ✓ os processos que os transformam.

3-Modelagem do Processo

Modelagem de dados transformado para conseguir implementar uma função do negócio.

Descrições do processamento são criadas para:

- ✓ adicionar,
- ✓ modificar,
- ✓ descartar ou
- ✓ recuperar um objeto de dados.

4-Geração da Aplicação

O RAD - uso de técnicas de quarta geração
- ✓ reutilização de componentes de programa
- ✓ cria componentes reusáveis.

São usadas ferramentas
Ex: Clarion, Delphi, Visual Basic, Asp.net, NetBeans, Eclipse entre outras...

5-Teste e Modificação

RAD enfatiza o reuso, muitos componentes já estão testados, isso reduz o tempo total de teste.

Mesmo assim, novos componentes devem ser testados e todas as interfaces devem ser exaustivamente exercitadas.

RAD

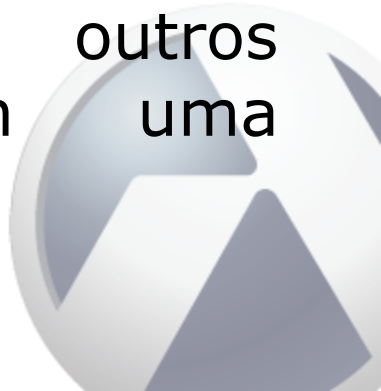
Caracteristicas:

- ✓ Permite desenvolvimento rápido / prototipagem de aplicações;
- ✓ Ciclos de desenvolvimento curtos (entre 60 e 90 dias);
- ✓ Cada função principal - direcionada para uma equipe RAD separada e então integrada a formar um todo;
- ✓ Criação e reutilização de componentes;
- ✓ Usado principalmente para aplicações de sistemas de informações;

RAD

Caracteristicas:

- ✓ Visibilidade mais cedo (protótipos);
- ✓ Maior flexibilidade (desenvolvedores podem reprojetar praticamente a vontade);
- ✓ Grande redução de codificação manual (wizards…);
- ✓ Envolvimento maior do usuário;
- ✓ Provável custo reduzido(tempo é dinheiro e também devido ao reuso);
- ✓ Aparência padronizada (As APIs e outros componentes reutilizáveis permitem uma aparência consistente).



RAD

O RAD é apropriado quando:

- ✓ A aplicação é do tipo "stand alone";
- ✓ Pode-se fazer uso de classes pré-existentes (APIs);
- ✓ A distribuição do produto é pequena;
- ✓ O escopo do projeto é restrito;
- ✓ O sistema pode ser dividido em vários módulos independentes;
- ✓ A tecnologia necessária tem mais de um ano de existência.
- ✓ Recomendado a equipes grandes (exige pessoal suficiente para criar várias equipes RAD)

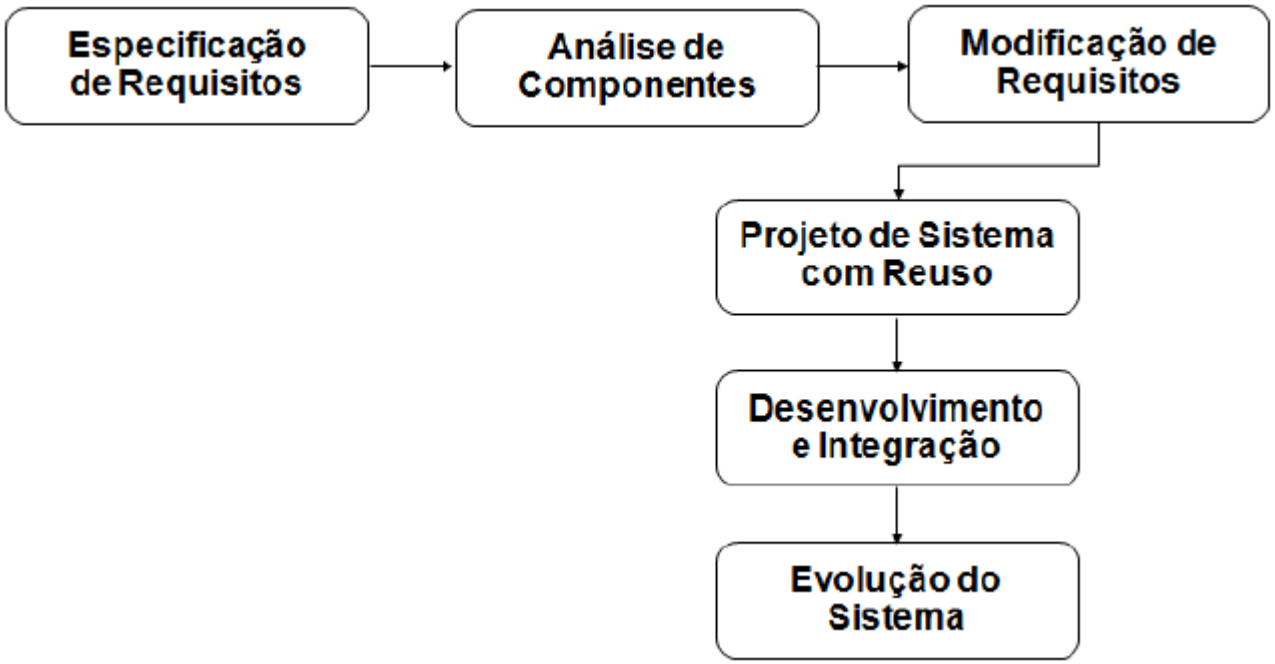
Desvantagens:

- ✓ Se uma aplicação não puder ser modularizada
- ✓ O envolvimento com o usuário tem que ser ativo;
- ✓ Riscos técnicos são altos e não é indicada quando se está testando novas tecnologias ou quando o novo software exige alto grau de interoperabilidade com programas de computador existentes.
- ✓ Funções desnecessárias (reuso de componentes);
- ✓ Problemas legais;
- ✓ Requisitos podem não se encaixar (conflitos entre desenvolvedores e clientes)

Orientado a Reuso

Orientado a Reuso

São baseados no uso sistemático de componentes/itens pré-existentes



Orientado a Reuso

Reuso:

Significa uso repetido, novo uso, reutilização.

Orientado a Reuso

Reutilização de software é o processo de incorporar/aproveitar itens em um novo produto:

- ✓ um novo código/funcionalidades (já existente)
- ✓ especificações de requisitos e projeto
- ✓ planos de teste, planos de teste,
- ✓ qualquer produto gerado em desenvolvimentos anteriores,

Benefícios:

- ✓ Melhoria de produtividade
- ✓ Aumento da qualidade, mais confiáveis, consistentes e padronizados
- ✓ Facilidade de manutenção e evolução
- ✓ Redução de Custo por reaproveitamento
- ✓ Redução de Custos por tempo de desenvolvimento

Requisitos para reutilização

- ✓ Componentes reutilizáveis bem documentados
- ✓ Qualidade dos Componentes
- ✓ Componentes criados com a finalidade de reuso
- ✓ Componentes que permitam adaptação a nova situação

Pontos de Atenção

- ✓ Identificação, recuperação e modificação de artefatos reutilizáveis - Biblioteca
- ✓ Ferramentas de apoio
- ✓ Barreiras legais e econômicas
- ✓ Necessidade da criação de incentivos à reutilização

Pós-Graduação

Engenharia de Software

Faculdade IMPACTA TECNOLOGIA

Obrigada

Marta Fuzioka

mrtfuzioka@uol.com.br