Projeto e Desenvolvimento de Software

Prof. Ronaldo C. de Oliveira, Dr.

ronaldo.co@ufu.br

UFU - 2018

Processo de Software

Processos de Software

- Conjunto coerente de atividades para especificar, projetar, implementar e testar sistemas de software

Objetivos

- Apresentar os modelos de processo de software
- Descrever os diferentes modelos de processos e quando eles podem ser utilizados
- Descrever em formas gerais os modelos de processo para engenharia de requisitos, desenvolvimento de software, testes e evolução
- Apresentar a tecnologia CASE para apoiar atividades do processo de software

O processo de software

- Um conjunto estruturado de atividades exigidas para desenvolver um sistema de software
  - Especificação
  - Projeto
  - Validação
  - Evolução
- Um modelo de processo de software é uma representação abstrata de um processo. Ele apresenta uma descrição de um processo a partir de uma perspectiva específica

Descrições de processo de software

- Quando descrevemos e discutimos processos, geralmente falamos sobre as atividades desses processos, tais como especificação de modelo de dados, desenvolvimento de interface de usuário, etc. e organização dessas atividades.
- Descrições de processos também podem incluir:
  - ✓ Produtos, que são os resultados de uma atividade do processo;
  - ✓ Papéis, que refletem as responsabilidades das pessoas envolvidas no processo;
  - ✓ Pré e pós-condições, que são declarações que são verdadeiras antes e depois de uma atividade do processo ser executada, ou um produto produzido.

Processos dirigidos a planos e ágeis

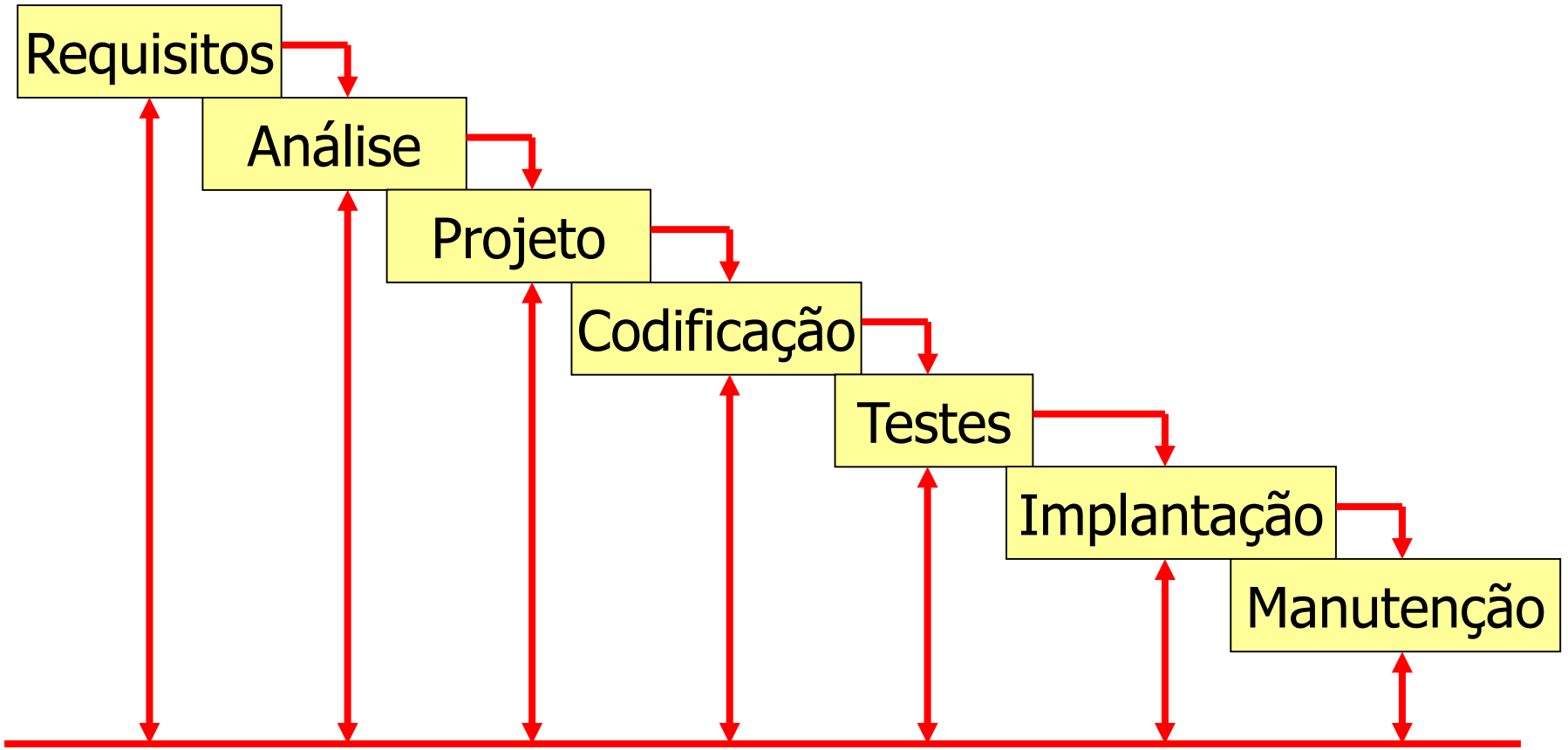
- Processos dirigidos a planos são processos em que todas as atividades do processo são planejadas com antecedência e o progresso é medido em relação a esse plano.
- Nos processos ágeis o planejamento é incremental e é mais fácil modificar o processo para refletir alterações nos requisitos do cliente.
- Na realidade, os processos mais práticos incluem elementos dos processos ágeis e dirigidos a planos.
- Não existe processo de software certo ou errado.

Modelos genéricos de modelos de processo de software

- O modelo cascata
  - Fases de especificação e desenvolvimento separadas e distintas
- Desenvolvimento evolucionário
  - Especificação e desenvolvimento são interfoliadas
- Desenvolvimento baseado em reuso
  - O sistema é montado a partir de componentes existentes

OBS: Na realidade a maioria dos grandes sistemas são desenvolvidos usando um processo que incorpora elementos de todos esses modelos.

Modelo Cascata - Waterfall



Requisitos do Sistema

- Identificação das necessidades e coleta inicial de dados;
- Estudo de viabilidade do sistema (técnica e econômica);
- Especificação e validação dos requisitos junto aos clientes;

Análise do Sistema

- Especificação do "o que" será o sistema, modelando todos os requisitos levantados de forma a definir o sistema:
  - Análise Estruturada Moderna
    - Modelo Essencial
  - Análise Orientada a Objetos
    - UML - Unified Modelling Language (Linguagem Unificada de Modelagem)

Projeto do Sistema

- Especificação de "como" o sistema será desenvolvido. Define:
  - Projeto de Interface
  - Projeto de Modelo de Dados
  - Projeto de Arquitetura
  - Projetos de Componentes
  - Projetos de Algoritmos

Codificação do Sistema

- Construção do sistema baseado em todos os documentos definidos e criados nas etapas de Análise e Projeto do Sistema;
- Etapa de maior esforço dentro do desenvolvimento de sistemas;

Testes do Sistema

- Depuração do código desenvolvido com a remoção dos erros de implementação
- Verificação e Validação permitem mostrar que o sistema está de acordo com a especificação e cumpre os requisitos do consumidor.
- Testes onde o sistema é executado em casos derivados da especificação e com processamento de dados reais.
- Esta etapa trabalha na garantia da qualidade do sistema desenvolvido.

Implantação do Sistema

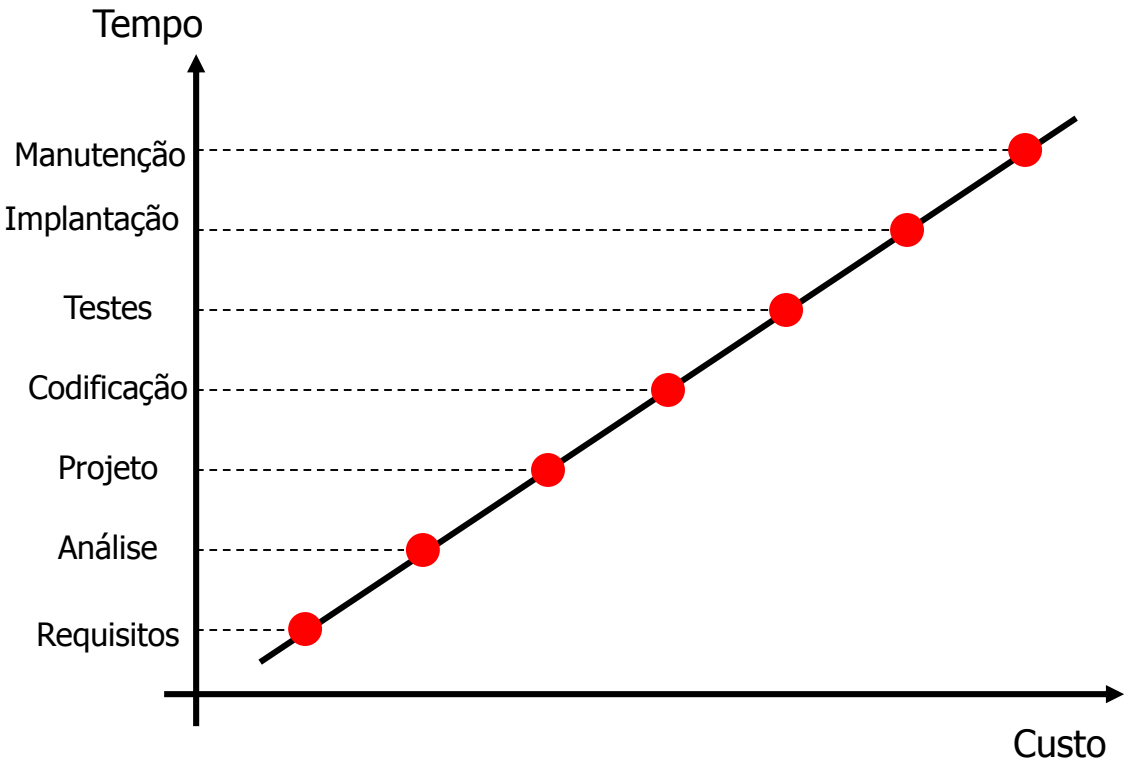
- Integração do Sistema - processo de "juntar" o hardware, software e as pessoas;
- Instalação do Sistema no ambiente operacional
- Treinamento dos usuários – essencial para garantir a aceitação do sistema;

Manutenção do Sistema

- Faz com que o sistema funcione no ambiente de negócios;
- Correção de erros e falas do sistema;
- Adequação do sistema ao usuários;
- Etapa muito importante para fidelizar os clientes;
- Deve ser realizado por profissionais capacitados.

Manutenção do Sistema

- Tipos de Manutenção:
  - Manutenção corretiva
  - Manutenção Adaptativa
  - Manutenção Perfectiva
  - Manutenção Evolutiva



Problemas do Modelo Cascata

- O principal inconveniente do modelo cascata é a dificuldade de acomodação de mudanças depois que o processo já foi iniciado. Em princípio, uma fase precisa ser completada antes de se mover para a próxima fase.

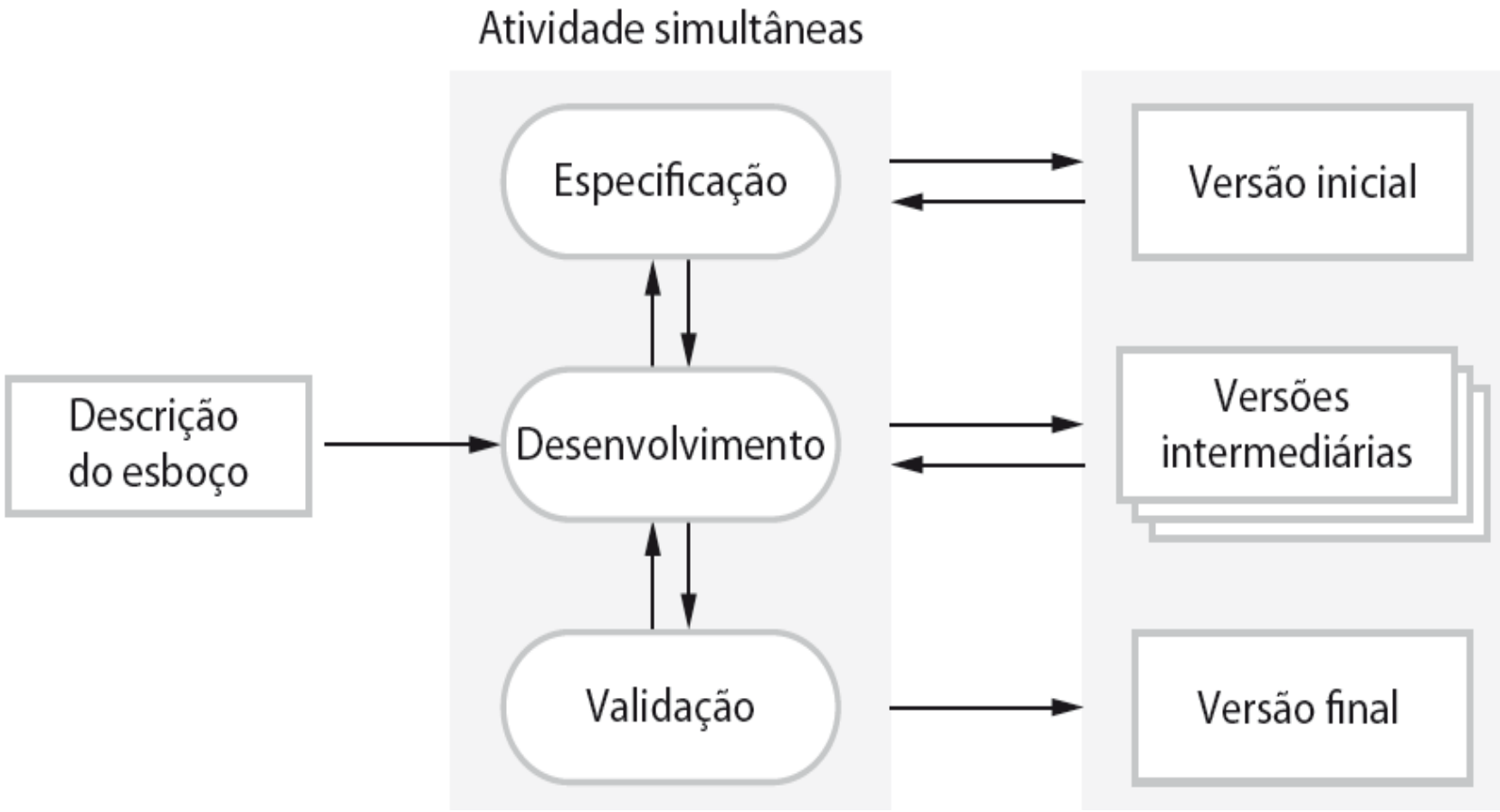
Problemas do Modelo Cascata

- Divisão inflexível do projeto em estágios distintos torna difícil responder às mudanças nos requisitos do cliente.
  - Por isso esse modelo só é apropriado quando os requisitos são bem entendidos e as mudanças durante o processo de projeto serão limitadas.
  - Poucos sistemas de negócio possuem requisitos estáveis.
- O modelo cascata é mais usado em projetos de engenharia de grandes sistemas onde o sistema é desenvolvido em vários locais.
  - Nessas circunstâncias, a natureza do modelo cascata dirigida a planos ajuda a coordenar o trabalho.

Desenvolvimento evolucionário

- Desenvolvimento exploratório
  - O objetivo é trabalhar com os clientes e evoluir um sistema final a partir de uma especificação genérica inicial. O desenvolvimento se inicia com as partes do sistema que estão compreendidas
- Fazer protótipos descartáveis
  - O objetivo é compreender os requisitos do sistema. O protótipo se concentra em fazer experimentos com partes dos requisitos que estejam mal compreendidas

Desenvolvimento evolucionário



Benefícios do Desenvolvimento evolucionário

- O custo para acomodar mudanças nos requisitos do cliente é reduzido.
  - A quantidade de análise e documentação que precisa ser feita é bem menor do que o necessária no modelo cascata.
- É mais fácil obter feedback do cliente sobre o trabalho de desenvolvimento que tem sido feito.
  - Os clientes podem comentar demonstrações do software e ver quanto foi implementado.
- Possibilidade de mais rapidez na entrega e implantação de software útil para o cliente.
  - Os clientes podem usar e obter ganhos do software mais cedo do que é possível no processo cascata.

Problemas do Desenvolvimento evolucionário

- O processo não é visível.
  - Gerentes precisam de entregas regulares para medir o progresso. Se os sistemas são desenvolvidos de forma rápida, não é viável do ponto de vista do custo produzir documentação para refletir todas as versões do sistema.
- A estrutura do sistema tende a degradar conforme novos incrementos são adicionados.
  - A menos que tempo e dinheiro sejam gastos na reconstrução para melhorar o software, as mudanças regulares tendem a corromper a estrutura do sistema. A incorporação posterior de mudanças no software se torna progressivamente mais difícil e cara.

Desenvolvimento de protótipos

- Pode ser baseado em linguagens ou ferramentas de prototipagem rápida.
- Pode deixar a funcionalidade de fora do teste.
  - A prototipação deve focar em áreas do produto que não são bem entendidas;
  - A checagem de erros e recuperação podem não estar incluídas no protótipo;
  - O foco deve ser em requisitos funcionais ao invés de não funcionais como por exemplo, a confiabilidade e a segurança.

Descarte de protótipos

- Os protótipos devem ser descartados depois do desenvolvimento, pois não são uma boa base para um sistema em produção:
  - Pode ser impossível ajustar o sistema para alcançar requisitos não funcionais;
  - Geralmente os protótipos não possuem documentação;
  - Geralmente a estrutura do protótipo é degradada por mudanças rápidas;
  - Provavelmente o protótipo não irá alcançar os padrões normais de qualidade organizacional.

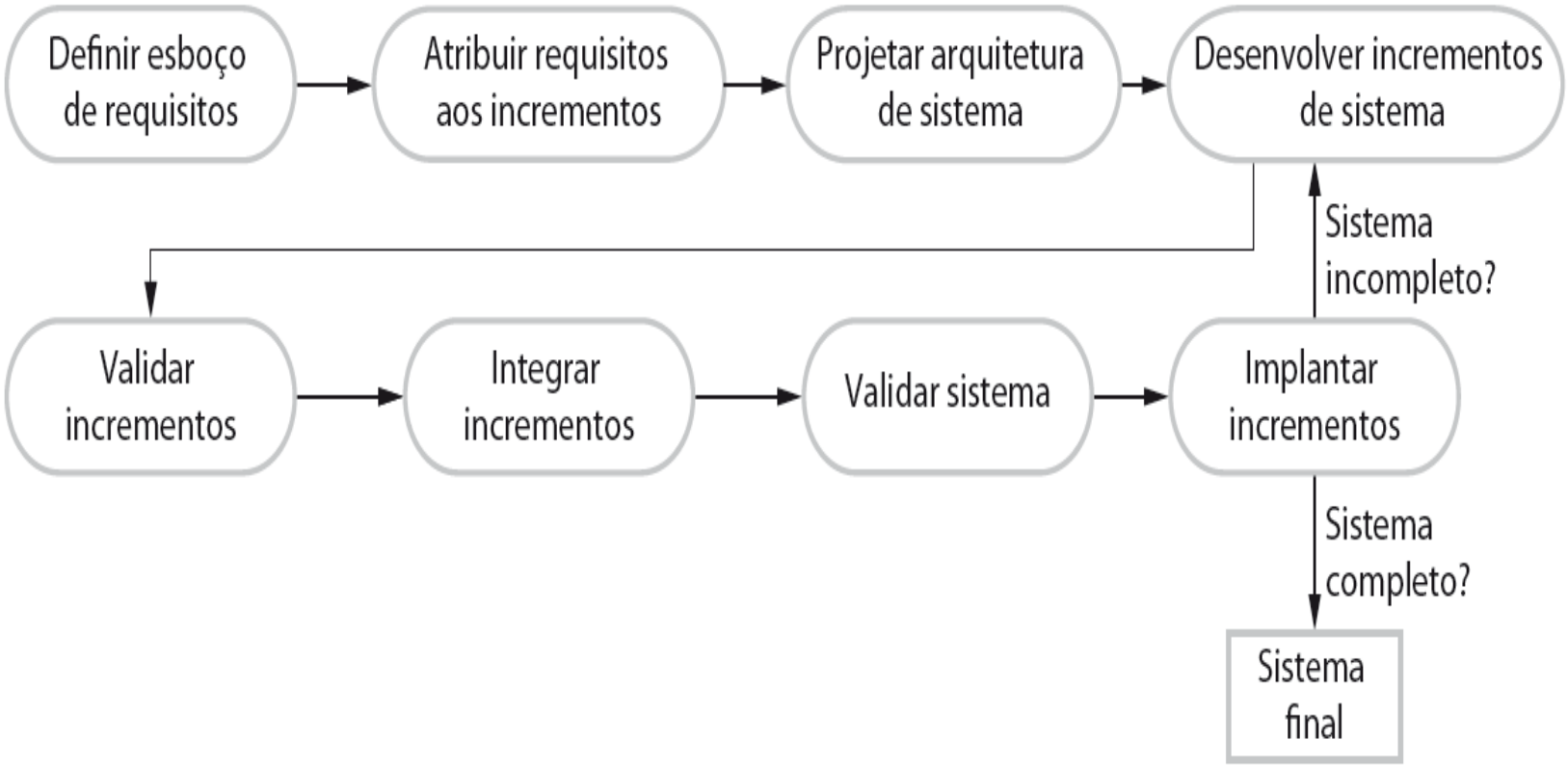
Entrega incremental

- Ao invés de entregar o sistema em uma única entrega, o desenvolvimento e a entrega são distribuídos em incrementos, nos quais cada incremento entrega parte da funcionalidade necessária.
- Os requisitos do usuário são priorizados e os requisitos de mais alta prioridade são incluídos nos primeiros incrementos.
- Assim que o desenvolvimento de um incremento é iniciado os requisitos são congelados, mas os requisitos dos incrementos posteriores podem continuar a evoluir.

Desenvolvimento e entrega incremental

- Desenvolvimento incremental
  - Desenvolve o sistema em incrementos e avalia cada incremento antes de proceder com o desenvolvimento do próximo incremento;
  - Abordagem normalmente usada em métodos ágeis;
  - Avaliação feita por representantes do usuário/cliente.
- Entrega incremental
  - Implanta um incremento para uso do usuário-final;
  - Avaliação mais realística sobre o uso prático do software;
  - Difícil de implementar para sistemas substitutos devido aos incrementos possuírem menos funções do que o sistema que está sendo substituido.

Entrega incremental



Vantagens da entrega incremental

- Os valores podem ser entregues ao cliente junto com cada incremento, e funções do sistema ficam disponíveis mais rápidamente.
- Primeiros incrementos agem como protótipos para ajudar a deduzir requisitos para incrementos posteriores.
- Menor risco de falha geral do projeto.
- Os serviços mais prioritários do sistema tendem a serem mais testados.

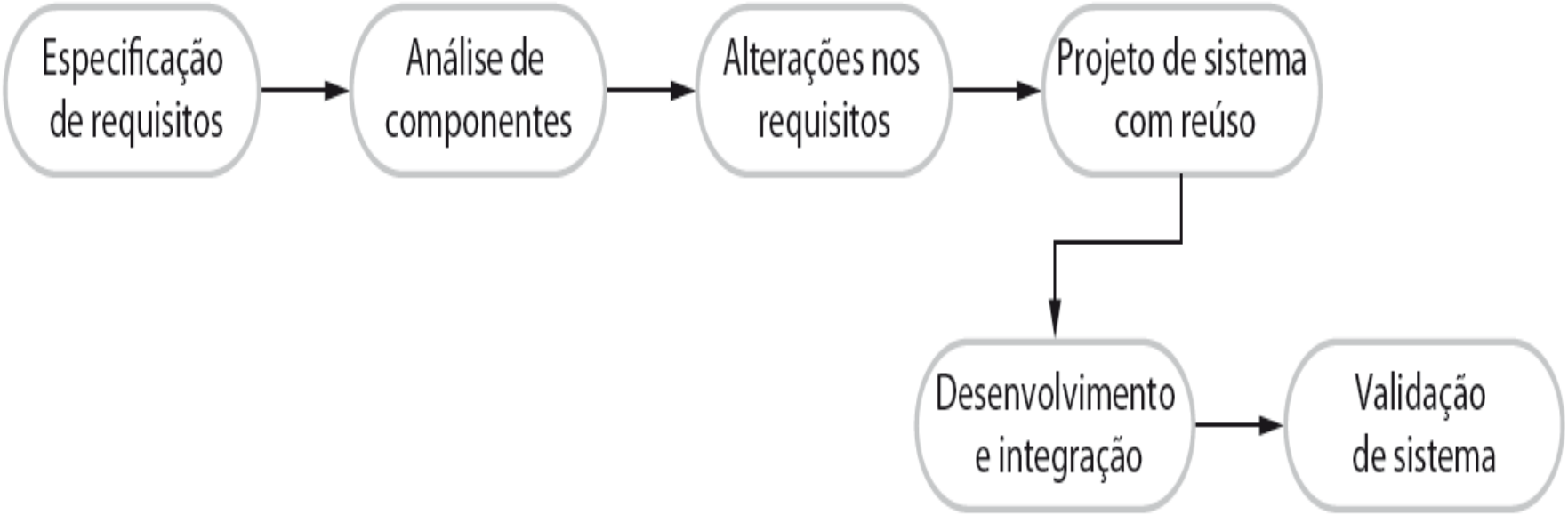
Problemas da entrega incremental

- A maioria dos sistemas requer um conjunto de funções básicas que são usadas por diferentes partes do sistema.
  - Como os requisitos não são definidos em detalhes até que um incremento seja implementado, pode ser difícil identificar funções comuns que são necessárias a todos os incrementos.
- A essência dos processos iterativos é que a especificação seja desenvolvida em conjunto com o software.
  - No entanto, essa pode entrar em conflito com o modelo de aquisição de muitas organizações, nos quais a especificação completa do sistema é parte do contrato de desenvolvimento do sistema.

Engenharia de software orientada a reúso

- Baseada no reúso sistemático em que os sistemas são integrados com componentes existentes ou sistemas COTS (Commercial-off-the-shelf).
- Estágios do processo:
  - Análise de componentes;
  - Modificação de requisitos;
  - Projeto de sistema com reúso;
  - Desenvolvimento e integração.
- Atualmente, o reúso é a abordagem padrão para a construção de vários tipos de sistemas de negócio.

Engenharia de software orientada a reúso



Tipos de componente de software

- Web services que são desenvolvidos de acordo com padrões de serviço e ficam disponíveis para chamada remota.
- Coleções de objetos que são desenvolvidas como um pacote para ser integrado com um framework como .NET ou J2EE.
- Sistemas de software stand-alone (COTS) que são configurados para uso em ambientes específicos.

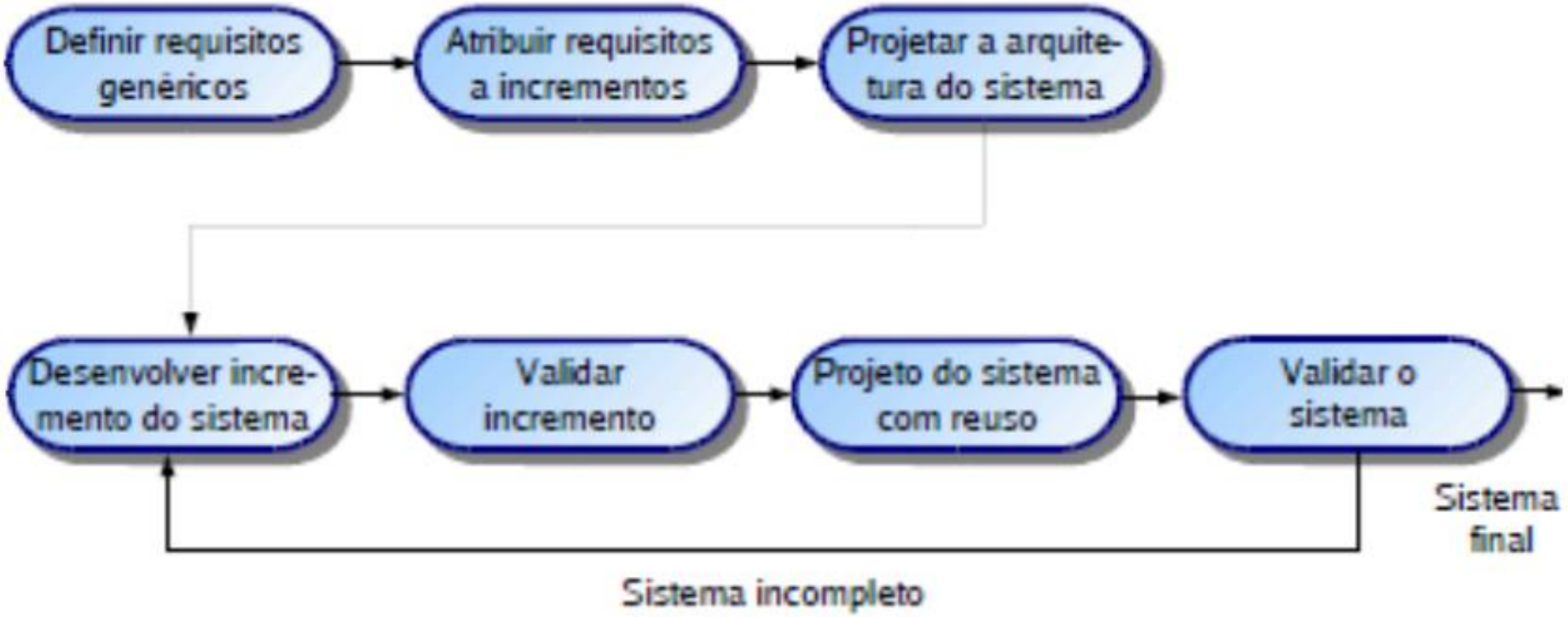
Iteração de processo

- Os requisitos do sistema SEMPRE evoluem ao longo de um projeto, portanto a iteração do processo, onde estágios iniciais são retrabalhados, é sempre parte do processo para sistemas grandes
- A iteração pode ser aplicada a qualquer um dos modelos genéricos de processo
- Duas abordagens (relacionadas)
  - Desenvolvimento incremental
  - Desenvolvimento espiral

Desenvolvimento incremental

- Ao invés de entregar o sistema como uma única entrega, quebram-se o desenvolvimento e a entrega em incrementos, com cada incremento entregando parte da funcionalidade requerida
- Os requisitos do usuário são priorizados e os requisitos de prioridade mais alta são incluídos nos incrementos iniciais
- Uma vez que o desenvolvimento de um incremento é iniciado, os requisitos são congelados, ainda que os requisitos para incrementos posteriores continuem a evoluir

Desenvolvimento incremental



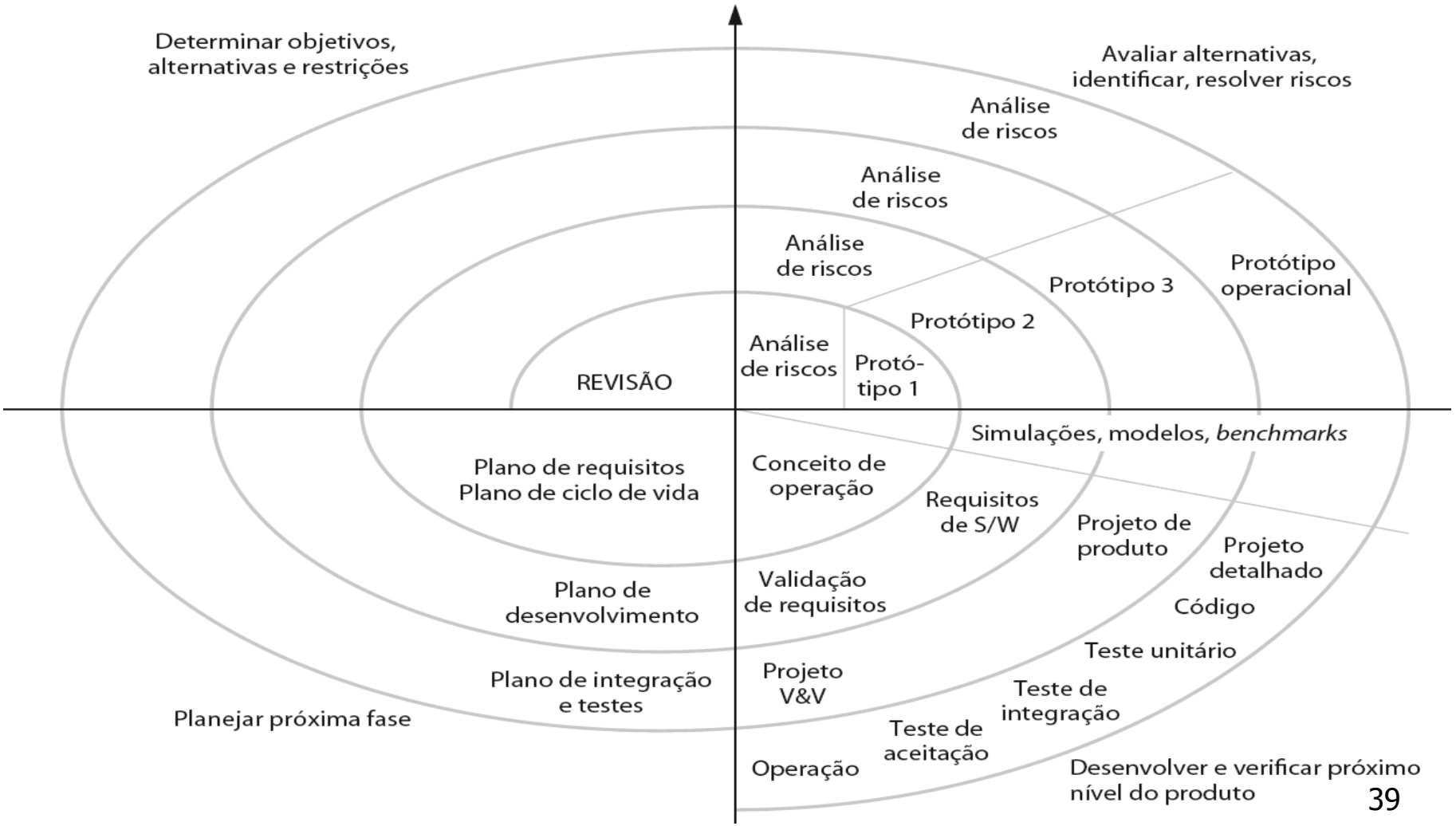
Vantagens do desenvolvimento incremental

- Cada incremento pode entregar valor para o cliente, portanto a funcionalidade do sistema está disponível mais cedo
- Incrementos iniciais atuam como um protótipo para ajudar a descobrir requisitos para os incrementos posteriores
- Menor risco de falha do projeto como um todo
- Os serviços de mais alta prioridade do sistema tendem a receber a maior parte dos testes

Modelo espiral de Boehm

- O processo é representado como uma espiral ao invés de uma sequência de atividades com retornos.
- Cada loop na espiral representa uma fase do processo.
- Não existem fases fixas como especificação ou projeto – os loops na espiral são escolhidos de acordo com a necessidade.
- Os riscos são avaliados explicitamente e resolvidos no decorrer do processo.

O modelo de processo de software espiral de Boehm



Setores do modelo espiral

- Definição de objetivos
  - São identificados os objetivos específicos para cada fase.
- Avaliação e redução de riscos
  - Os riscos são avaliados e atividades executadas para reduzir os principais riscos.
- Desenvolvimento e validação
  - Um modelo de desenvolvimento para o sistema é escolhido, pode ser qualquer um dos modelos genéricos.
- Planejamento
  - O projeto é revisto e a próxima fase da espiral é planejada.

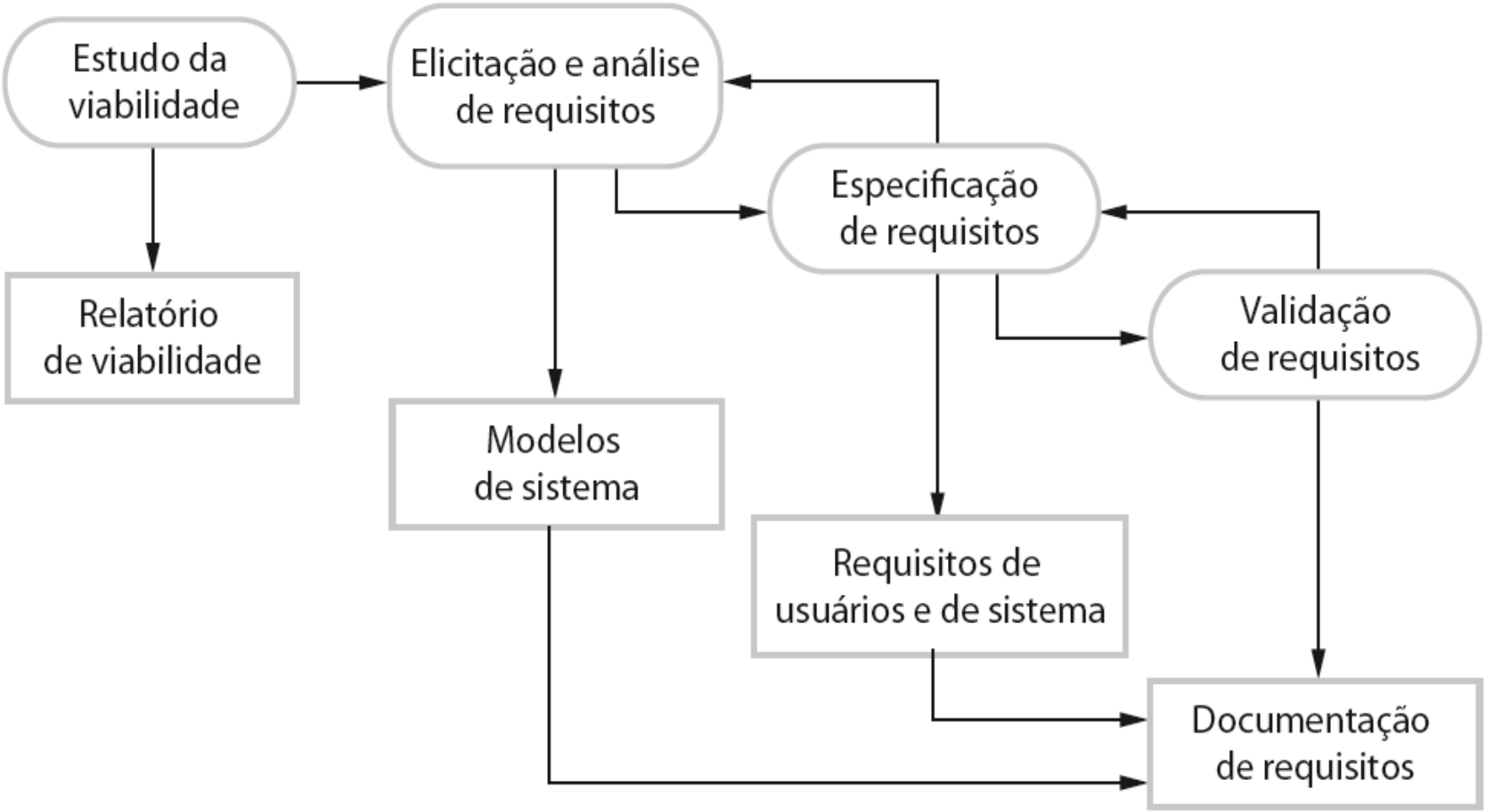
Atividades do processo

- Processos de software reais são sequências intercaladas de atividades técnicas, colaborativas e gerenciais com o objetivo geral de especificar, projetar, implementar e testar um sistema de software.
- As quatro atividades de processo básicas, **especificação**, **desenvolvimento**, **validação** e **evolução** são organizadas de forma diferente em processos de desenvolvimento distintos.
- No modelo cascata, elas são organizadas em sequências, enquanto no desenvolvimento incremental são intercaladas.

Especificações de software

- O processo de estabelecer quais serviços são necessários e as restrições na operação e desenvolvimento do sistema.
- Processo de engenharia de requisitos
  - Estudo de viabilidade
    - É técnica e financeiramente viável construir o sistema?
  - Elicitação e análise de requisitos
    - O que os stakeholers do sistema precisam ou esperam do sistema?
  - Especificação de requisitos
    - Definição dos requisitos em detalhes.
  - Validação de requisitos
    - Verificação da completude dos requisitos.

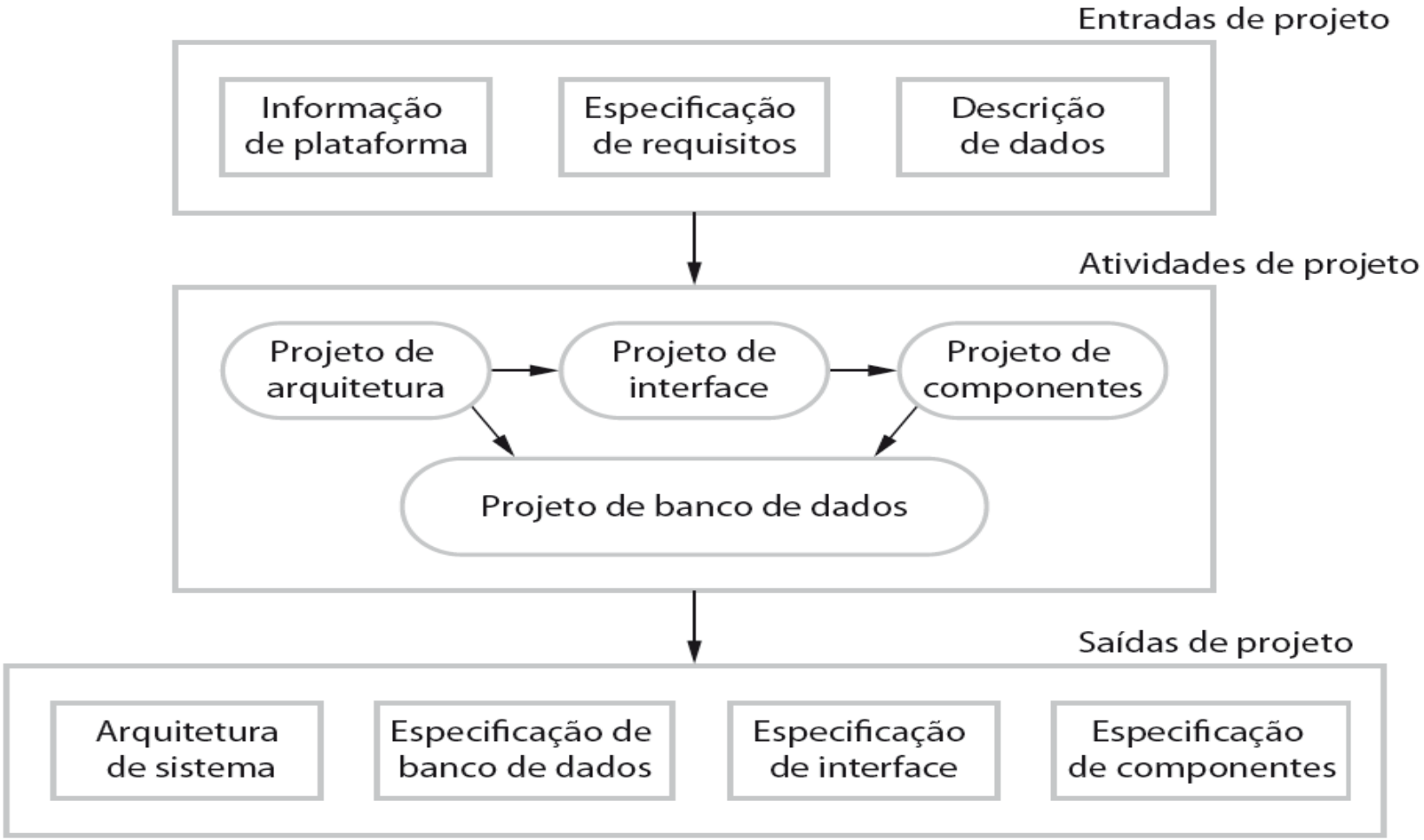
O processo de engenharia de requisitos



Projeto e implementação de software

- O processo de converter a especificação de sistema em um sistema executável.
- Projeto de software
  - Design de uma estrutura de software que materialize a especificação;
- Implementação
  - Transformar essa estrutura em um programa executável;
- As atividades de projeto e implementação são intimamente ligadas e podem ser intercaladas.

Modelo geral do processo de design

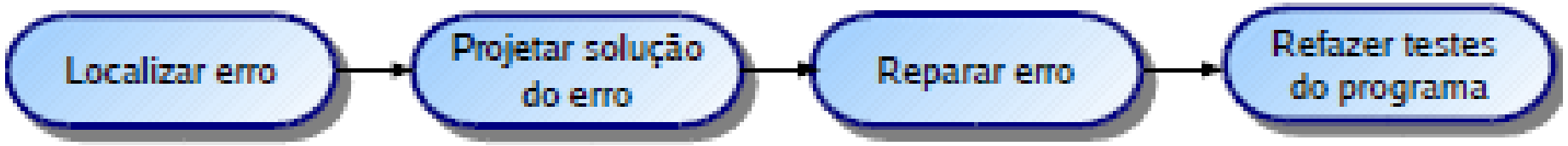


Atividades de projeto

- Projeto de arquitetura, em que você identifica a estrutura geral do sistema, os componentes principais (as vezes chamados sub-sistemas ou módulos), seus relacionamentos e como são distribuídos.
- Projeto de interface, em que você define as interfaces entre os componentes do sistema.
- Projeto de componente, em que você projeta como cada componente do sistema irá operar separadamente.
- Projeto de banco de dados, em que você projeta as estruturas de dados do sistema e como essas serão representadas no banco de dados.

Programação e depuração (*debugging*)

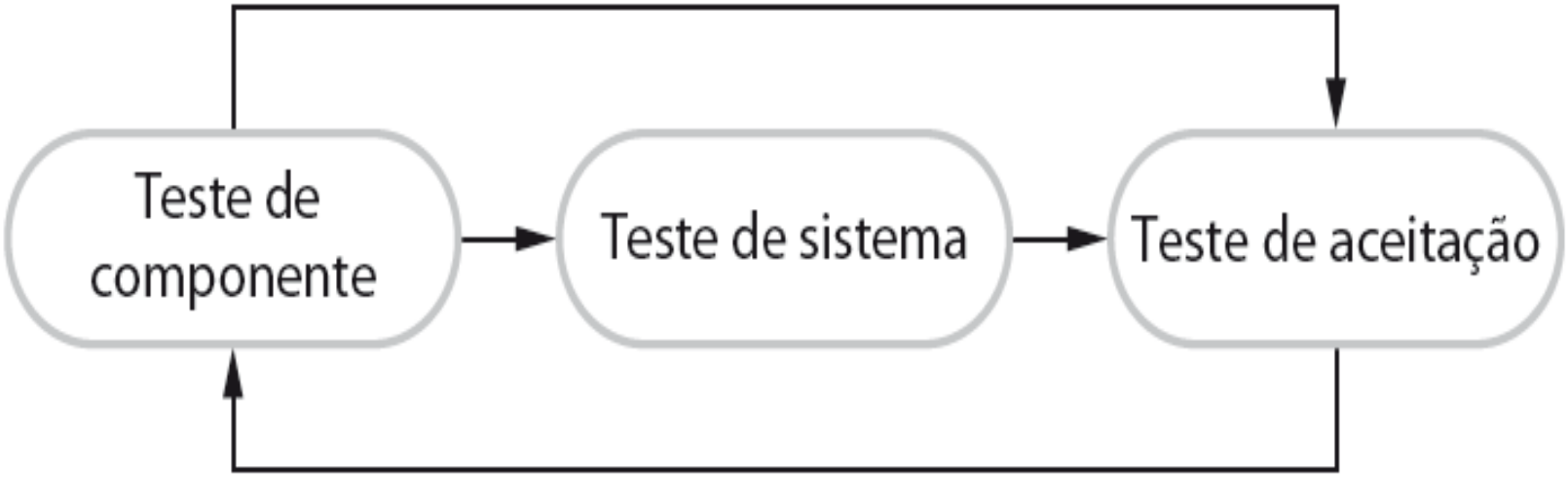
- Traduzir um projeto em um programa e remover erros do programa
- Programação é uma atividade pessoal . não há um processo genérico de programação
- Os programadores desenvolvem alguns testes do programa para descobrir falhas em um programa e remover essas falhas em um processo de depuração



Validação de software

- Verificação e validação (V & V) serve para mostrar que o sistema está em conformidade com sua especificação e está de acordo com os requisitos do cliente.
- Envolve processos de inspeção e revisão, e testes do sistema.
- Testes do sistema envolvem executar o sistema com casos de teste. São provenientes de especificações dos dados reais que deverão ser processados pelo sistema.
- O teste é a atividade de V & V mais usada.

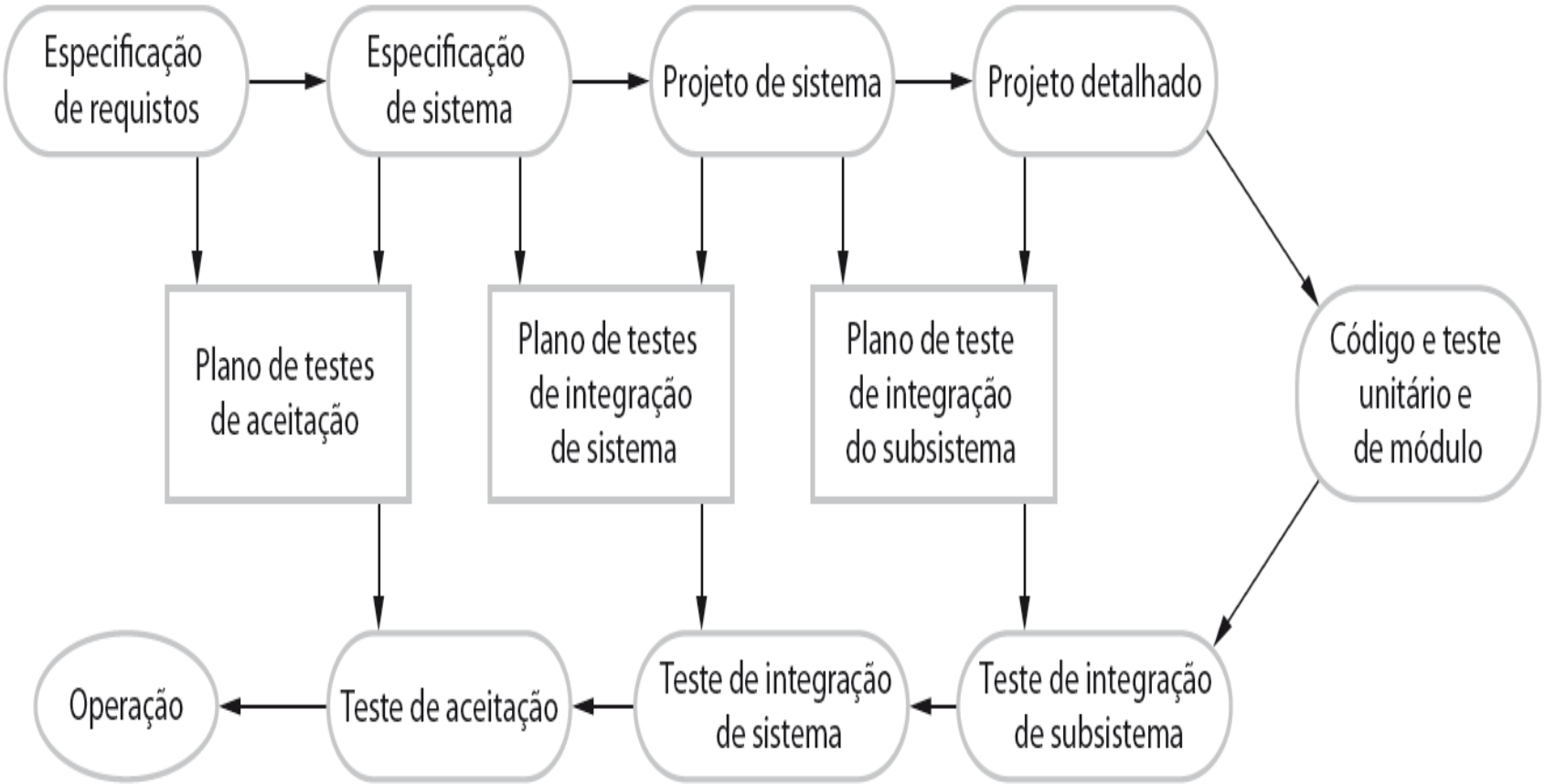
Estágios de teste



Estágios de teste

- Teste de desenvolvimento ou de componente
  - Componentes individuais são testados independentemente;
  - Componentes podem ser funções ou objetos , ou agrupamentos coerentes dessas entidades.
- Teste de sistema
  - Teste do sistema como um todo. Teste de propriedades emergentes são particularmente importantes.
- Teste de aceitação
  - Teste com dados do cliente para checar se o sistema está de acordo com as necessidades do cliente.

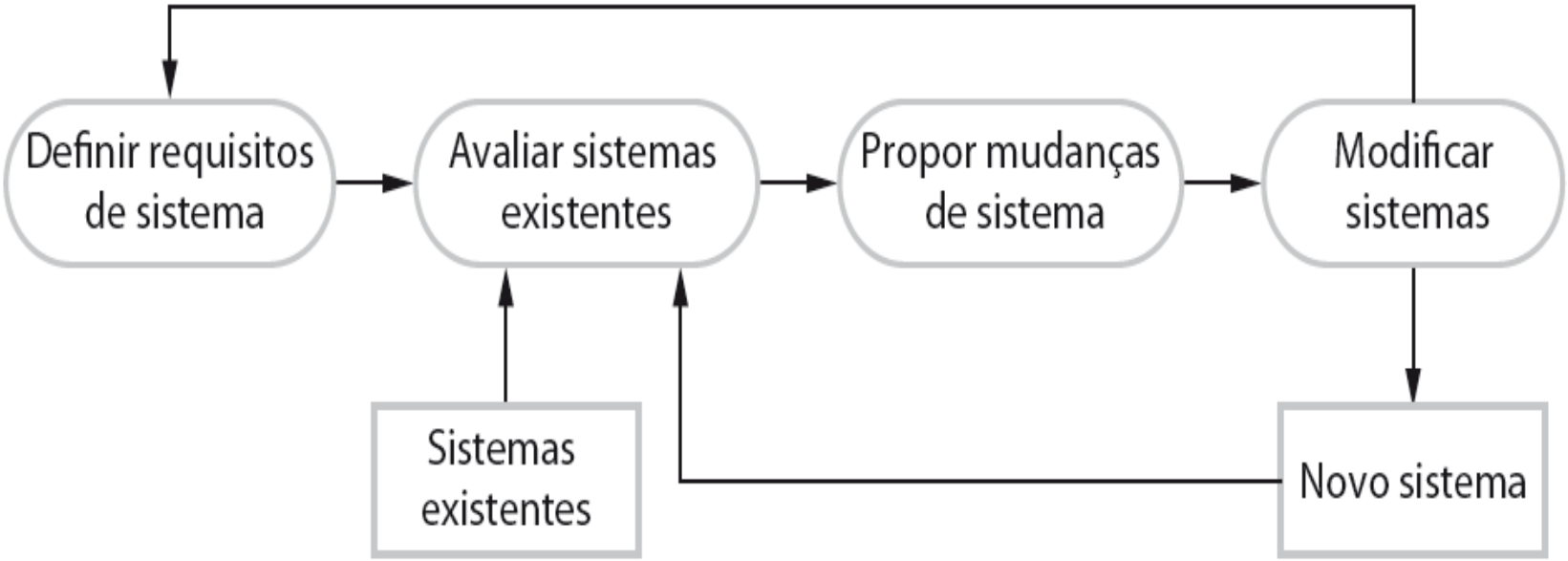
Fases de teste em um processo de software dirigido a planos



Evolução do software

- Os softwares são inerentemente flexíveis e podem mudar.
- Conforme os requisitos mudam, conforme mudam as circustâncias do negócio, o software que dá suporte ao negócio também deve evoluir e mudar.
- Apesar de ter acontecido uma demarcação entre desenvolvimento e evolução (manutenção) essa precisa se tornar cada vez mais irrelevante já que tem diminuído a quantidade de sistemas completamente novos.

Evolução do sistema



Pontos Importantes

- Os processos de software são as atividades envolvidas na produção de um sistema de software. Os modelos de processo de software são representações abstratas desses processos.
- Modelos de processo gerais descrevem a organização dos processos de software. Exemplos desses processos gerais incluem o modelo 'cascata', desenvolvimento incremental e desenvolvimento orientado a reúso.
- A engenharia de requisitos é o processo de desenvolver uma especificação de software.

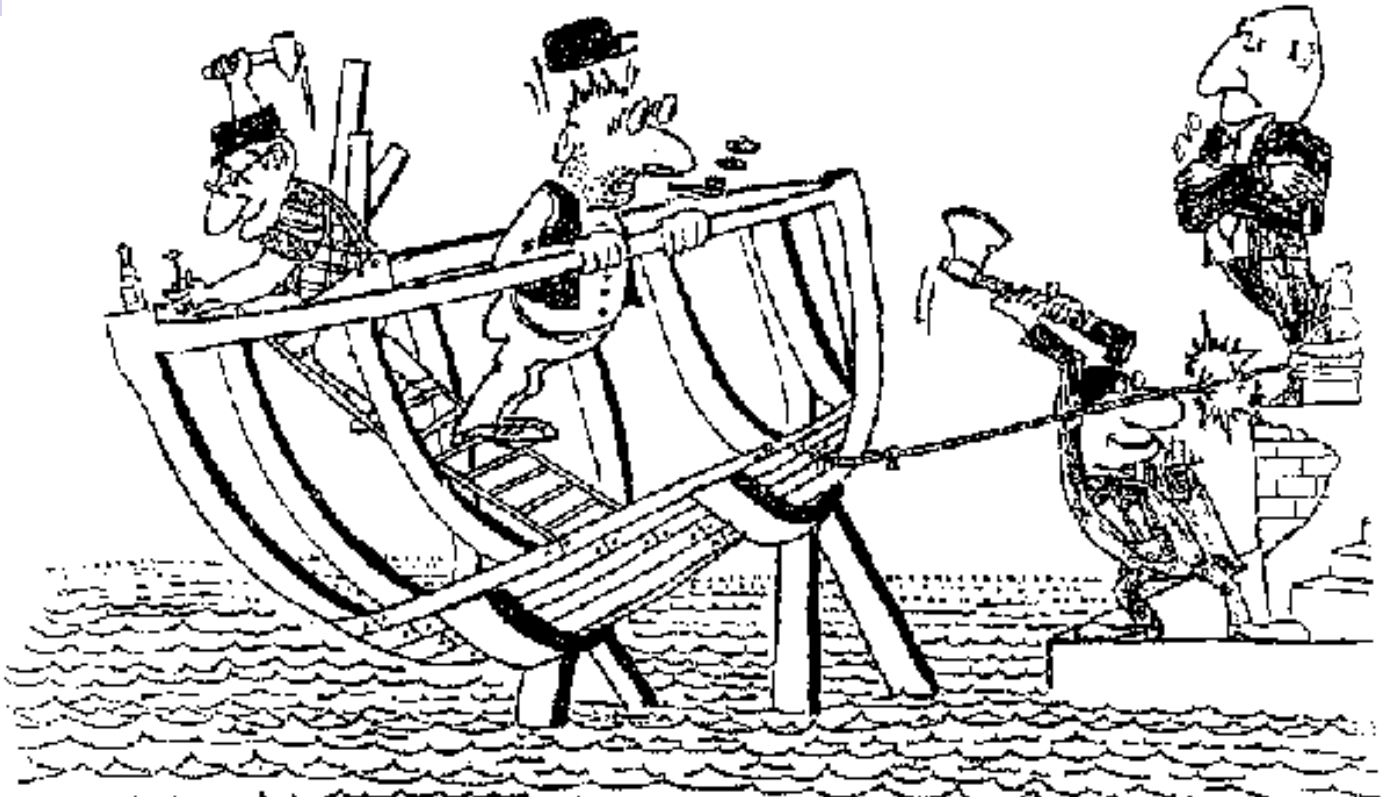
Pontos Importantes

- Processos de projeto e implementação se preocupam em transformar uma especificação de requisitos em um sistema de software executável.
- A validação de software é o processo de checar se o sistema está em conformidade com sua especificação e se esse está de acordo com as necessidades reais do usuário do sistema.
- A evolução de software ocorre quando você altera sistemas de software existentes para adequá-los a novas necessidades. O software precisa evoluir para continuar útil.

Atividade

Grandes verdades sobre o desenvolvimento de sistemas!!!

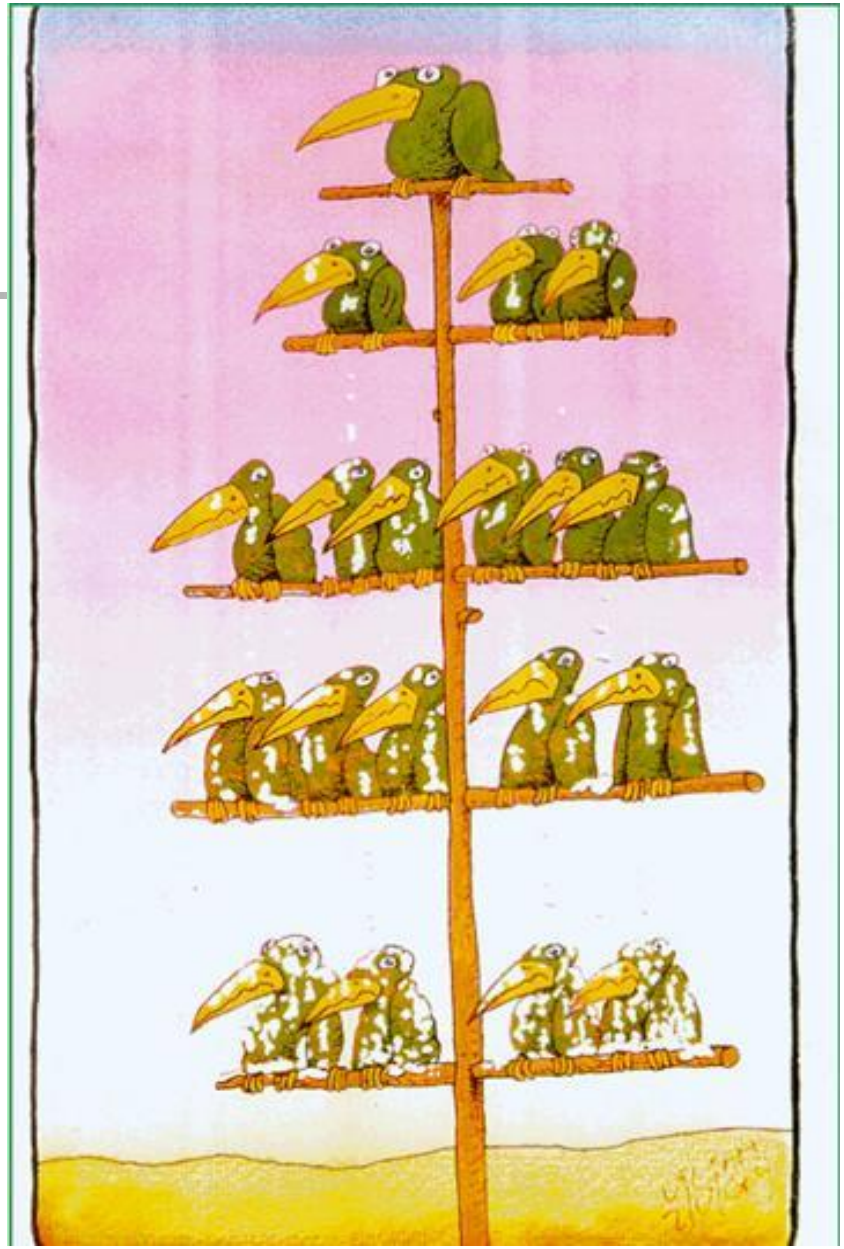
Não estabeleça prazos audaciosos demais



Prazo é prazo !

Nem toda apresentação será um sucesso

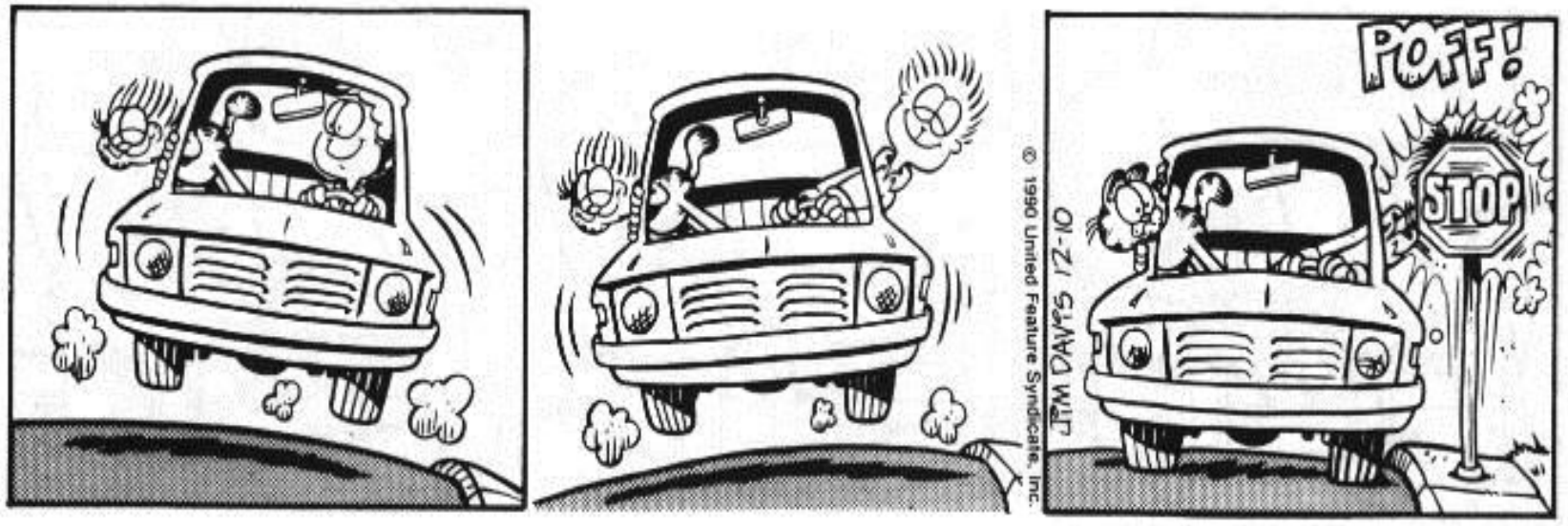
A estrutura hierárquica tradicional só atrapalha



Preste atenção aos sinais do mercado

Escolha atributos significativos para seu cliente

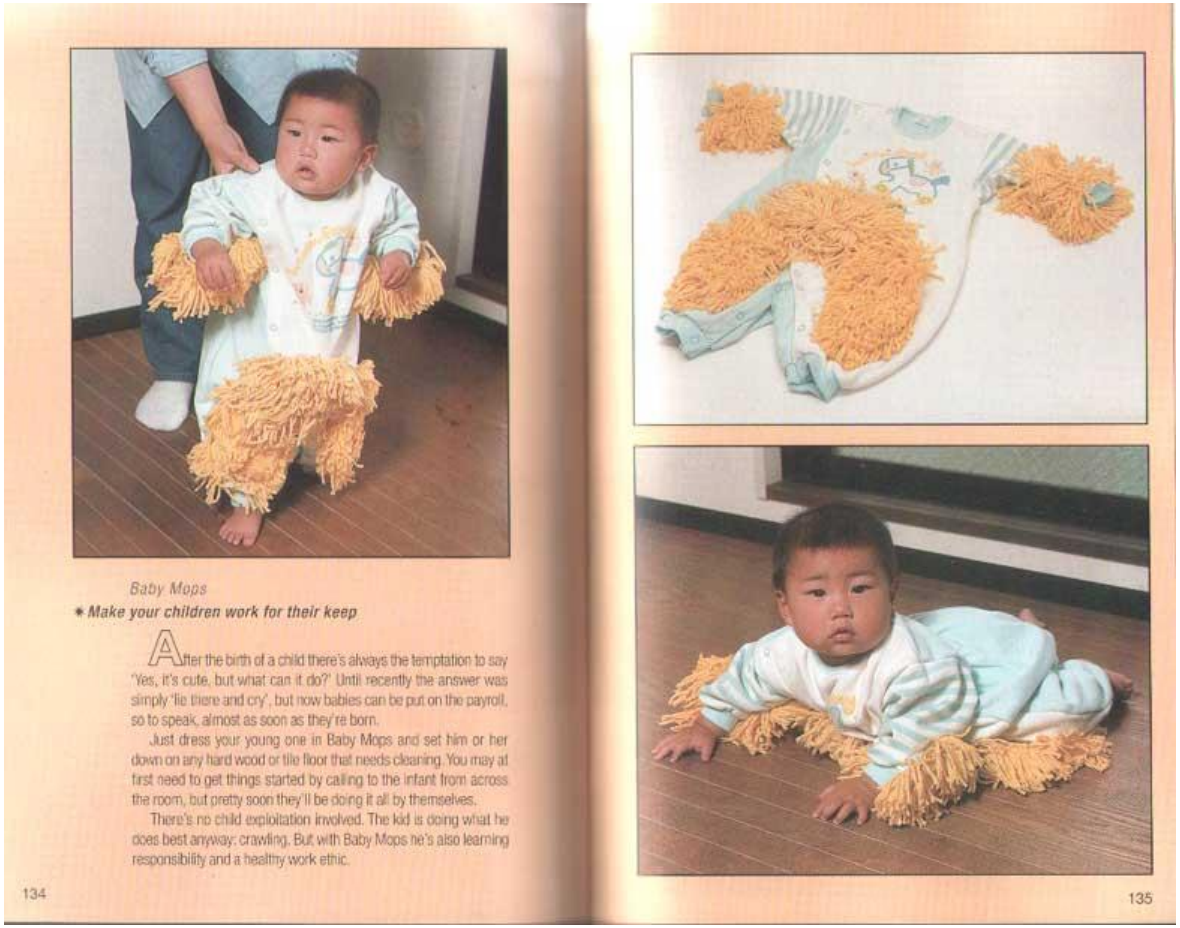
O que serve para um cliente pode não servir para o próximo



Nada pode parar a automação

Busque soluções eficientes

Seja inovador; aproveite as oportunidades



Baby Mops

*** Make your children work for their keep**

After the birth of a child there's always the temptation to say 'Yes, it's cute, but what can it do?' Until recently the answer was simply 'lie there and cry', but now babies can be put on the payroll, so to speak, almost as soon as they're born.

Just dress your young one in Baby Mops and set him or her down on any hard wood or tile floor that needs cleaning. You may at first need to get things started by calling to the infant from across the room, but pretty soon they'll be doing it all by themselves.

There's no child exploitation involved. The kid is doing what he does best anyway: crawling. But with Baby Mops he's also learning responsibility and a healthy work ethic.

134

135

Escolha a ferramenta mais adequada para cada situação

Soluções técnicas nem sempre podem ser implantadas

Ajuda on-line pode ser útil ...

Enfermeira, acesse a internet, vá até www.cirurgia.com e clique no ícone "O que fazer quando você está totalmente perdido".

Mas não acredite em tudo que vem pela Internet

O bom da Internet é que ninguém sabe que você é um cachorro...

Experiência em simulações pode ser de serventia

Previsão e realização nem sempre saem como planejado

O uso de soluções tecnológicas é inevitável...

Mas as dificuldades das pessoas devem ser consideradas...

E não esqueça do treinamento dos usuários

Acostume-se a trabalhar sob pressão

Acredite em você mesmo. Tenha confiança.

Vá até o fim!