



# Requisitos

Engenharia, Levantamento, Elicitação, Gerenciamento

Fernando Pedrosa – [fpedrosa@gmail.com](mailto:fpedrosa@gmail.com)

# Bibliografia

- ▶ **Sommerville, Ian. Software Engineering. Editora: Addison Wesley. (capítulos sobre Requisitos)**

# Engenharia de Requisitos

- ▶ Área da Engenharia de Software preocupada com:
  - Objetivos do mundo real para sistemas de software
  - Funções de sistemas de software
  - Restrições em sistemas de software
  - E o relacionamento entre esses fatores

**Objetivando especificações precisas do comportamento do SW e sua evolução**

# Objetivos

- ▶ Estabelecer e manter concordância com os clientes e outros envolvidos sobre o que o sistema deve fazer
- ▶ Oferecer aos desenvolvedores do sistema uma compreensão melhor dos requisitos do sistema
- ▶ Definir (delimitar) as fronteiras do sistema

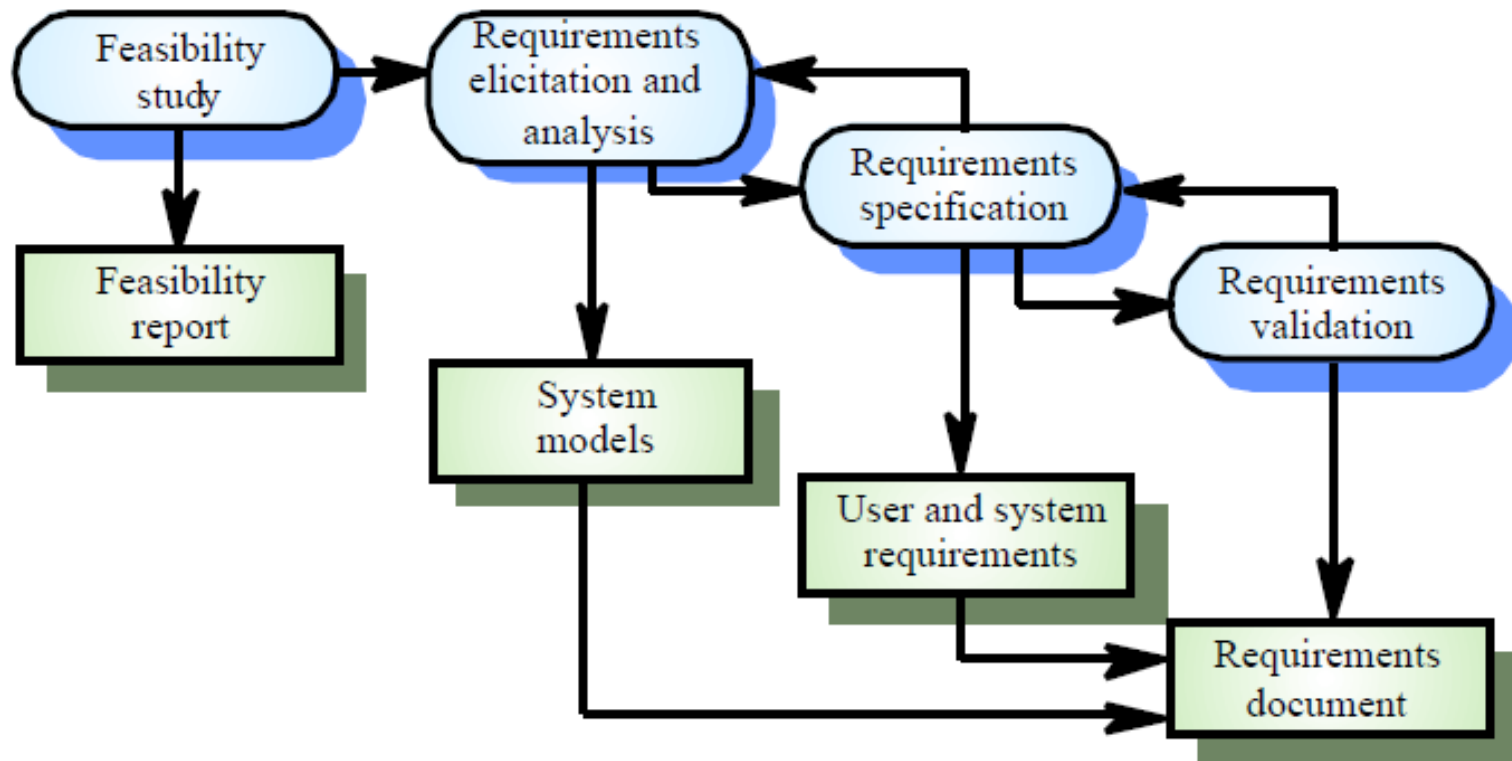
# Objetivos

- ▶ Fornecer uma base para planejar o conteúdo técnico das iterações.
- ▶ Fornecer uma base para estimar o custo e o tempo de desenvolvimento do sistema.
- ▶ Definir uma interface de usuário para o sistema, focando nas necessidades e metas dos usuários

# Características

- ▶ Queremos ter uma especificação de requisitos que seja:
  - Completa
  - Consistente
  - Não ambígua
  - Correta
- ▶ Que sirva, inclusive, de base para um acordo entre as partes envolvidas no processo de desenvolvimento de software

# O processo de Engenharia de Requisitos



# Exercícios [1]

**(SAD/PE – CESPE 2010)**

[35-a] A engenharia de requisitos de um software, em geral, precede a engenharia dos requisitos do sistema de informações no qual o software será usado.

**(SERPRO – CESPE 2010)**

[78] A área de atividade de requisitos de software apresenta maior interface com a engenharia de sistemas quando comparada à área de análise e projeto de software.

[79] Visando à maior efetividade no processo de desenvolvimento, os requisitos de software geralmente são, em geral, desenvolvidos antes dos requisitos do sistema.

# Exercícios [1]

(ANAC – CESPE 2009)

[65] Requisitos descrevem um acordo ou contrato entre duas partes, especificando, entre outros aspectos, o que o sistema de software deve fazer para ser aprovado em um teste de aceitação.

# O que é um requisito?

- ▶ Um requisito é definido como “uma condição ou uma capacidade com a qual o sistema deve estar de acordo”
  - Pode ser desde uma indicação abstrata, de alto nível, até uma especificação matemática detalhada

**Em resumo: definem o que o sistema deve fazer e sob quais limitações ele é requisitado a operar**

# Exemplos de Requisitos

- ▶ “O sistema deve ser capaz de debitar e creditar uma conta corrente”
- ▶ “O sistema deve ser capaz de realizar transferências bancárias do tipo DOC e TED”
- ▶ “O sistema deve suportar pelo menos 20 transações por segundo”
- ▶ “O sistema deve estar disponível, pelo menos, durante 10 horas por dia”

# Partes interessadas (*stakeholders*)

- ▶ Engenheiros de SW responsáveis pelo desenvolvimento do sistema
- ▶ Usuários finais que irão usar o sistema depois de entregue
- ▶ Especialistas de domínio que possuem informações sobre os processos atuais
- ▶ Fiscais externos, que verificam se o sistema satisfaz os requisitos legais

# Problemas com Requisitos...

- ▶ Virem de várias fontes
- ▶ Não refletirem as reais necessidades dos usuários do sistema
- ▶ Serem inconsistentes e/ou incompletos
- ▶ Podem ter um alto custo para mudanças, depois de acordados
- ▶ Mal entendidos entre clientes e desenvolvedores

# Problemas com Requisitos...



Como o cliente explicou



Como o líder de projeto entendeu



Como o analista projetou



Como o programador codificou



Como o consultor de negócio descreveu



Como o projeto foi documentado



Como o produto foi instalado



Como cobraram do cliente



Como foi suportado

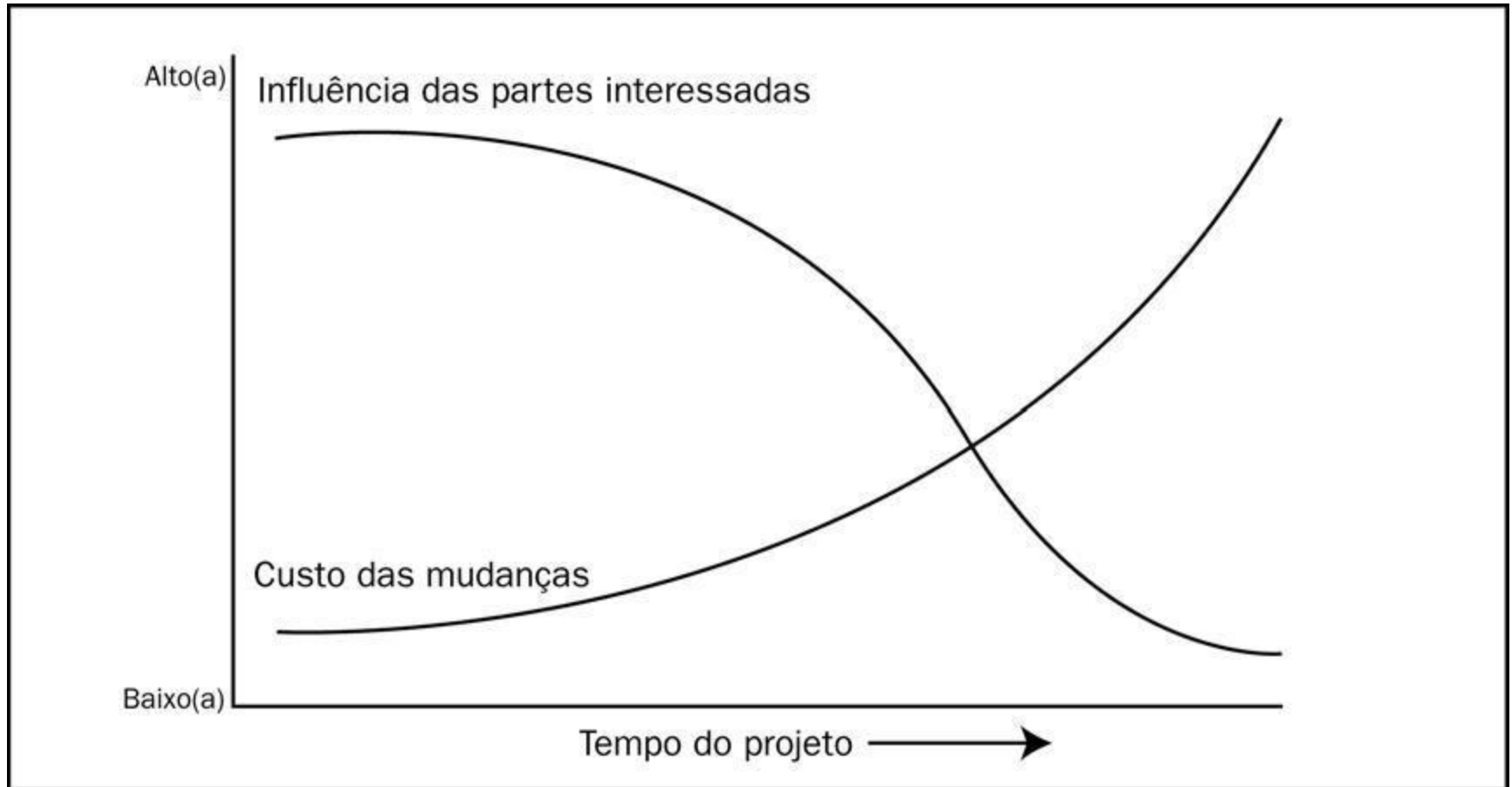


O que o cliente realmente precisava

# Importância da ER

- ▶ Quanto mais tarde problemas com requisitos são detectados, maior é o custo para corrigi-los
  - Pode custar até cinco vezes mais, caso o processo já esteja com ênfase na codificação ou até vinte vezes mais, caso esteja na manutenção
- ▶ **O sucesso das etapas posteriores depende da qualidade dos requisitos gerados**

# Custo da mudança...



# Exercícios [2]

INMETRO (CESPE 2010) 32 A engenharia de requisitos pode ser dividida em dois grupos de atividades: o desenvolvimento de requisitos e a gerência de requisitos. O desenvolvimento de requisitos inclui as seguintes etapas: elicitação de requisitos, análise e negociação de requisitos, especificação e modelagem de requisitos e validação de requisitos. A esse respeito, assinale a opção correta.

- A) Nas atividades de desenvolvimento de requisitos para um sistema, deve-se tentar reduzir a participação efetiva dos usuários do sistema, visto que ela gera mais problemas que contribuições positivas.
- B) Para a fase de especificação e modelagem de requisitos, a técnica mais recomendada é o JAD (joint application design), que, desenvolvido pela IBM, permite a criação de sistemas mais eficazes em menor tempo.

# Exercícios [2]

- C) A gerência de requisitos e o desenvolvimento de requisitos são atividades independentes uma da outra, por isso não é necessário haver interação das equipes que as realizam.
- D) Atualmente, as empresas não têm tido dificuldade para implantar as atividades de desenvolvimento de requisitos e de gerência de requisitos. De fato, essas atividades estão plenamente implantadas na quase totalidade das organizações e empresas de software.
- E) São atividades-chave para um gerenciamento de requisitos eficaz: analisar o problema, compreender as necessidades dos envolvidos, definir e refinar o escopo do sistema e gerenciar as mudanças de requisitos.

# Tipos de Requisitos

- ▶ Funcionais
  - Definem funcionalidades do sistema
- ▶ Não Funcionais
  - Expressam restrições sob as quais o sistema deve operar ou qualidades específicas que o software deve ter
- ▶ De Domínio
  - Vêm do domínio da aplicação do sistema e refletem características do domínio

# Tipos de Requisitos

- ▶ Requisitos permanentes (estáveis)
  - Derivados da atividade principal da organização. Exemplo: em um hospital sempre haverá requisitos relativos aos médicos, aos pacientes, aos tratamentos, etc. Derivados do modelo de domínio
- ▶ Requisitos Voláteis
  - Requisitos que se modificam durante o desenvolvimento ou quando o sistema está em uso. Exemplo: Requisitos resultantes de políticas governamentais

# Tipos de Requisitos

Requisitos voláteis são divididos em:

- ▶ **Requisitos Mutáveis**
  - Se modificam por causa do ambiente do sistema
- ▶ **Requisitos emergentes**
  - Surgem à medida que a compreensão do cliente do sistema se desenvolve
- ▶ **Requisitos conseqüentes**
  - Resultam da introdução do sistema no ambiente do usuário
- ▶ **Requisitos de compatibilidade**
  - Dependem de outro equipamento ou processo. Conforme eles mudam, o requisito também muda

# Requisitos Funcionais

- ▶ Descrevem funcionalidades ou serviços do sistema
- ▶ Dependem do tipo de software, dos usuários e do contexto onde ele será utilizado
- ▶ Podem ser escritos em alto nível, se forem voltados ao cliente, ou podem ser especificados em detalhe, para desenvolvedores

# Requisitos Funcionais

## *Exemplos*

- ▶ O sistema deve permitir cadastrar os dados pessoais dos clientes
- ▶ O sistema deve emitir relatórios gerenciais
- ▶ O sistema deve permitir a baixa automática de estoque quando da venda um produto

# Requisitos não Funcionais

- ▶ Definem propriedades do sistema e restrições:
  - Usabilidade
  - Confiabilidade
  - Desempenho
  - Manutenibilidade
  - Escalabilidade
  - Portabilidade
  - ...

# Requisitos não Funcionais

- ▶ Requisitos do **processo** também podem ser especificados obrigando o uso de uma determinada ferramenta CASE, linguagem de programação ou processo de desenvolvimento
- ▶ Requisitos não funcionais podem ser, e normalmente são, mais críticos do que os requisitos funcionais

# Tipos de RNFs

## ▶ Requisitos do produto

- Requisitos que especificam que o software entregue deve se comportar de um determinado modo, por ex.: ser confiável, robusto, rápido, etc.

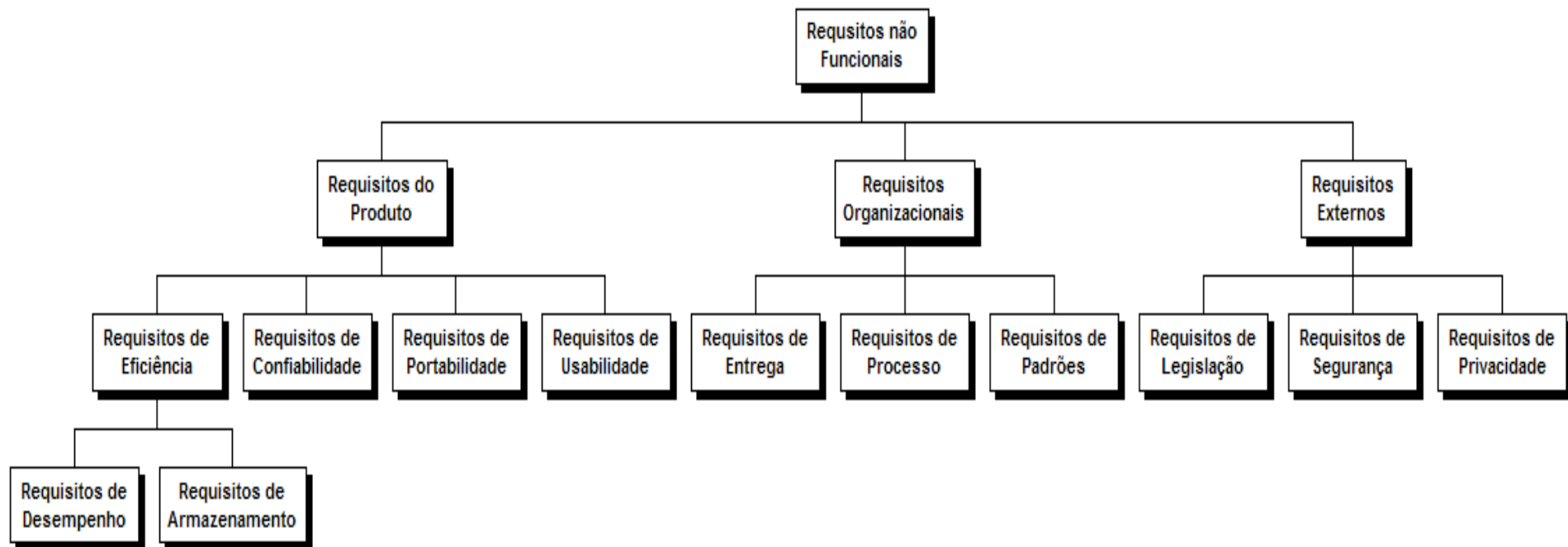
## ▶ Requisitos organizacionais

- Requisitos que são consequência das políticas e procedimentos organizacionais, como padrões, processos, etc.

## ▶ Requisitos externos

- Requisitos que são externos ao sistema e seu desenvolvimento, ex: legislação, interoperabilidade, etc.

# Tipos de RNFs



# Problemas com RNFs

- ▶ Requisitos não funcionais podem ser extremamente difíceis de especificar precisamente
  - Como verificamos RNFs?
- ▶ Requisitos não funcionais devem ser verificáveis
  - Usando alguma medida que possa ser objetivamente testada
- ▶ O problema é que, muitas vezes, RNFs são conflitantes entre si!

# Medindo RNFs

Propriedade	Medida
Desempenho	Transações por segundo; Tempo de resposta para eventos; etc.
Armazenamento	Megabytes; Número de chips ROM;
Usabilidade	Tempo de treinamento; Número de cliques de mouse;
Confiabilidade	Tempo médio entre falhas; Taxa de ocorrência de falhas; Disponibilidade;
Robustez	Tempo para recomeçar depois de uma falha; Probabilidade de corrupção de dados após falha;
Portabilidade	Porcentagem de declarações dependentes de plataforma; Número de plataformas-alvo

# Requisitos de Domínio

- ▶ São derivados do Domínio da aplicação e descrevem as características e funcionalidades do sistema que refletem o domínio em questão
- ▶ São transformados, posteriormente, em requisitos funcionais ou restrições (RNFs)
- ▶ Se não forem satisfeitos, também podem inviabilizar o funcionamento do sistema

# Requisitos de Domínio

## ▶ Exemplo

“A desaceleração do trem deve ser computada como:  $D_{trem} = D_{controle} + D_{gradiente}$  onde  $D_{gradiente} = 9.81 \text{ ms}^2$  vezes o gradiente compensado/alpha, onde os valores de  $9.81 \text{ ms}^2/\alpha$  variam de acordo com o tipo do trem”

## ▶ Que requisitos funcionais e não funcionais podemos derivar daí?

# Problemas com Req. de Domínio

- ▶ São expressados na linguagem do domínio da aplicação
  - Alta chance do engenheiro de software não entender
- ▶ Conhecimento tácito
  - Os especialistas de domínio entendem tão bem da sua área que, muitas vezes, não pensam em tornar os requisitos de domínio explícitos

# Exercícios [3]

(TRE/MT – CESPE 2010)

[31–a] O levantamento de requisitos tem como objetivo compreender o problema a ser resolvido e identificar necessidades. Os requisitos podem ser funcionais, que definem as funcionalidades do sistema, ou não funcionais, que não estão relacionados às funcionalidades.

(TRE/MT – CESPE 2010)

[33–a] Requisitos funcionais descrevem as propriedades emergentes do sistema, como segurança e tempo de resposta.

[33–b] Requisitos não funcionais são descritos de forma qualitativa e não quantitativa.

[33–c] Requisitos são provenientes de pessoas relevantes para o sistema, e não de outros sistemas que interagem com o sistema que está sendo especificado.

# Exercícios [3]

(ANA – ESAF 2009)

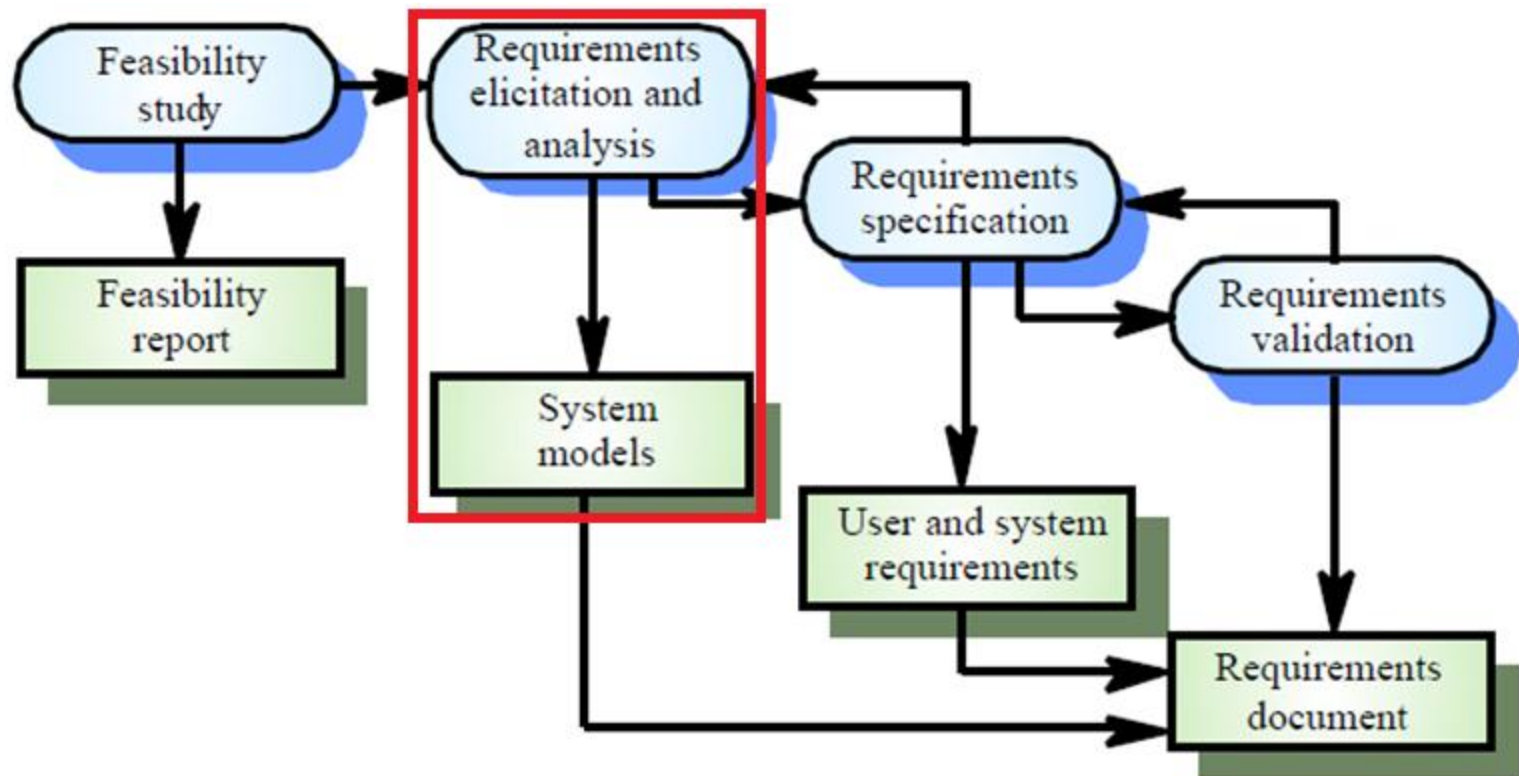
12– Analise as seguintes afirmações sobre requisitos de sistemas de software:

- I. Requisitos funcionais declaram as funções que o sistema deve fornecer, seu comportamento, e ainda, o que o sistema não deve fazer.
- II. Requisitos de domínio são, exclusivamente, funcionais, pois exibem as características do domínio de aplicação do sistema.
- III. Requisitos não-funcionais compreendem restrições sobre serviços ou funções do sistema.

Assinale a opção correta.

- a) Apenas as afirmações I e II são verdadeiras.
- b) Apenas as afirmações I e III são verdadeiras.
- c) Apenas as afirmações II e III são verdadeiras.
- d) As afirmações I, II e III são verdadeiras.

# Elicitação e Análise de Requisitos



# Elicitação: o que é?

- ▶ **Elicitar:** descobrir, tornar explícito, obter o máximo de informações para o conhecimento do objeto em questão
- ▶ A equipe técnica deve esclarecer:
  - O domínio da aplicação
  - Os serviços que a aplicação deve oferecer
  - As restrições sob as quais a aplicação deve operar
- ▶ Envolve vários *stakeholders*

# Problemas

- ▶ Os interessados não sabem o que querem
- ▶ Os interessados descrevem os problemas em sua própria linguagem
- ▶ Os requisitos de cada parte interessada podem ser conflitantes
- ▶ Fatores políticos e organizacionais podem influenciar os requisitos do sistema

# Atividades

- ▶ Entendimento do Domínio da Aplicação
  - Entender os problemas atuais na organização e como o software a ser implementado se ajustará a ela
- ▶ Descoberta (levantamento) dos Requisitos
  - Interagir com as partes interessadas para descobrir seus requisitos

# Técnicas de Elicitação

- ▶ Entrevistas
- ▶ Questionários
- ▶ Leituras de documentos
- ▶ Observações e análises sociais (etnografia)
- ▶ Workshops de requisitos
- ▶ Cenários (Casos de Uso)
- ▶ Prototipagem
- ▶ ...

# Técnicas de Elicitação: Etnografia

# Etnografia

- ▶ Técnica de observação utilizada para compreender os requisitos sociais e organizacionais
- ▶ Um cientista social se insere no ambiente de trabalho onde o sistema será implantado e analisa como as pessoas trabalham
- ▶ As pessoas não precisam explicar o seu trabalho
- ▶ Fatores sociais e organizacionais importantes podem ser observados

# Escopo da Etnografia

- ▶ Requisitos que são derivados da forma como as pessoas trabalham, e não de como os desenvolvedores **pensam** que os processos funcionam
- ▶ Requisitos que são derivados da cooperação e compreensão das atividades das outras pessoas

# Técnicas de Elicitação: Workshop de requisitos

# Workshop de requisitos

- ▶ Põe todos os *stakeholders* juntos por um período intensivo (focado)
- ▶ O **facilitador** de um workshop é o responsável pelas atividades logísticas e de organização, que inclui:
  - Dar a todos a oportunidade de falar
  - Manter a sessão sobre controle
  - Reunir informações para atributos de requisitos aplicáveis
  - Registrar as descobertas
  - Resumir a sessão e elaborar conclusões

# Workshop de requisitos

- ▶ Durante o Workshop, outras técnicas de identificação podem ser utilizadas
  - Brainstorming
  - Interpretação de papéis
  - Revisão de requisitos existentes, etc.
- ▶ Ao final do workshop, o facilitador resume as descobertas em um formato apresentável

# Exercícios [4]

**(SERPRO – CESPE 2010)**

[66] A entrevista é uma técnica de elicitação de requisitos simples, eficiente e direta, e por esses motivos, pode ser usada como fonte exclusiva de informação acerca dos requisitos do sistema.

**(TCE/RN – CESPE 2009)**

[51] A etnografia é uma técnica utilizada para a descoberta de requisitos de sistemas de software na qual, por meio de observações, procura-se compreender os requisitos sociais e organizacionais do ambiente onde o sistema será usado.

# Exercícios [4]

(Governo do ES – CESPE 2009)

[71] Durante a elicitação de requisitos de um projeto pode ser empregada uma técnica denominada workshop, na qual os principais stakeholders de um projeto são reunidos por um curto período de tempo. Essa técnica prevê a existência de um facilitador, que deve ser um dos stakeholders e não deve interferir nas decisões do grupo ou emitir opiniões.

# Técnicas de Elicitação: Casos de Uso

# Casos de Uso

- ▶ Desenvolvidos por I. Jacobson, são parte integrante da UML
- ▶ São descrições textuais das funcionalidades do sistema a partir da perspectiva do usuário
- ▶ Usados para mostrar quais funcionalidades o sistema oferece e que usuários se comunicam com ele



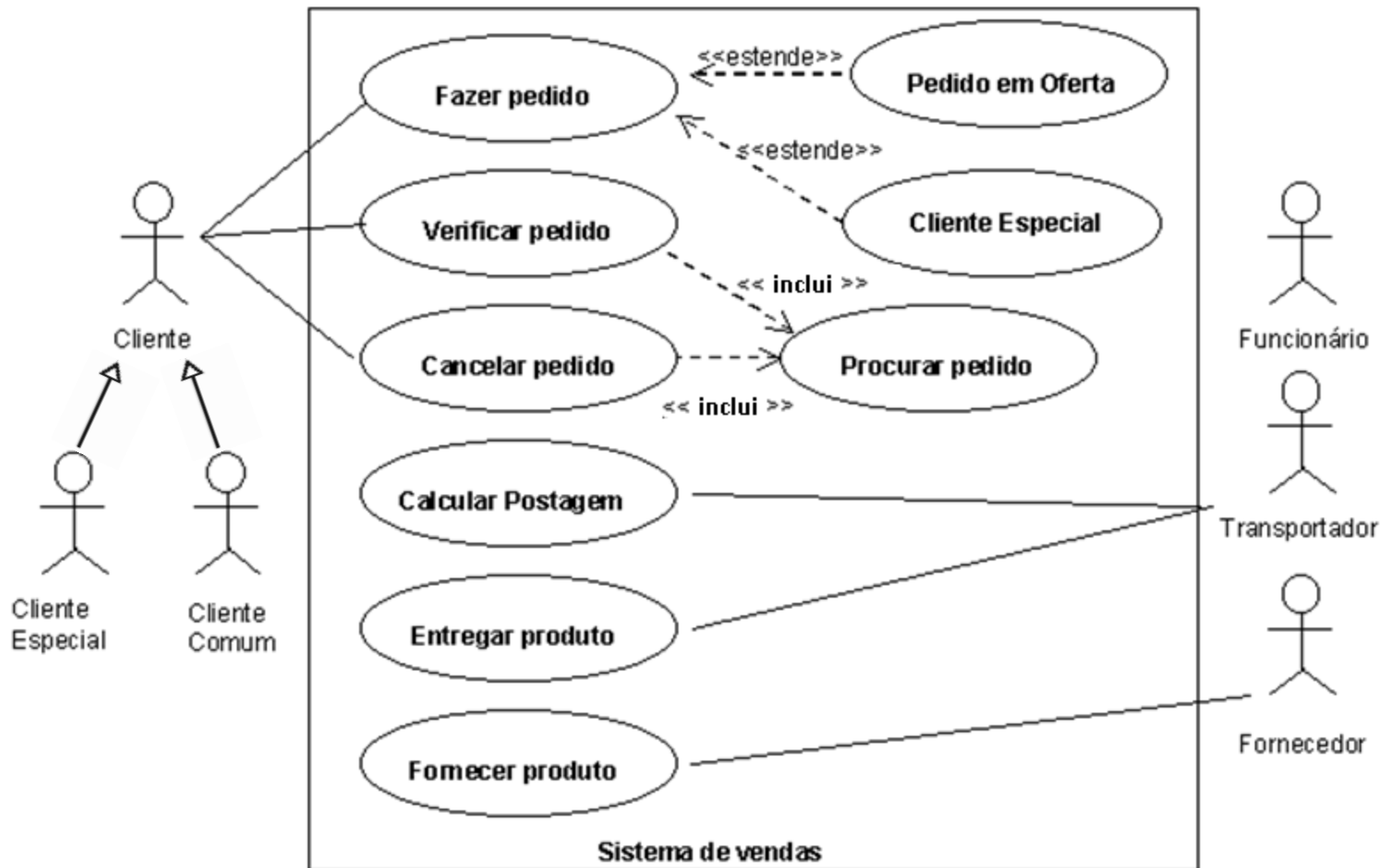
# Quem usa os Casos de Uso?

- ▶ Clientes e Usuários
- ▶ Arquitetos
- ▶ Analistas, Projetistas e Implementadores
- ▶ Testadores
- ▶ Gerentes
- ▶ Escritores da documentação
- ▶ ...

# Diagramas de Caso de Uso

- ▶ Servem para facilitar o entendimento de um sistema mostrando a sua “visão externa”
- ▶ São usados para modelar o contexto de um sistema ou um subsistema
- ▶ São uma das maneiras mais comuns de documentar requisitos do sistema
  - Delimitam o seu escopo
  - Definem suas funcionalidades

# Exemplo: Sistema de Vendas



# Especificação do Caso de Uso

**Nome:** Fazer Pedido

**Descrição:** Caso de uso que especifica o fluxo de ações para o cliente fazer um pedido no Sistema

**Atores:** Cliente

**Pré Condição:** O cliente deve estar logado no sistema

# Especificação do Caso de Uso

## Fluxo Principal de Eventos:

**P1.** O caso de uso começa quando o cliente seleciona a opção “Fazer Pedido”

**P2.** O cliente fornece seu nome e endereço e fornece o código do produto [EXT1]

**P3.** O sistema fornece a descrição e o preço do produto [INC1]

**P4.** O cliente fornece as informações de pagamento e escolhe a opção “confirmar” [A1]

**P5.** O sistema verifica as informações fornecidas e envia os dados para o sistema de pagamento [E1]

**P6.** O caso de uso é encerrado

# Especificação do Caso de Uso

## Pontos de Extensão

**EXT1.** O sistema estende o caso de uso “Pedido em Oferta”

## Pontos de Inclusão

**INC1.** O sistema inclui o caso de uso “Dar informação do produto”

## Fluxo Alternativo de Eventos

**A1.** No passo P4 cliente seleciona a opção “cancelar”

**A1.1** O sistema não grava o pedido e o fluxo retorna para o passo **P6**

# Especificação do Caso de Uso

## Fluxo Excepcional de Eventos

E1. No passo P5 o sistema verifica que as informações fornecidas estão incorretas

E1.1 O sistema pede ao cliente para corrigir as informações e o fluxo retorna ao passo P4

## Pós Condições

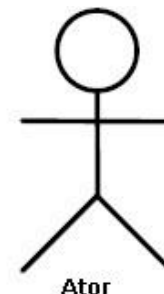
O pedido deve ter sido gravado no sistema e marcado como confirmado

# Especificação do Caso de Uso

- ▶ O Caso de Uso pode conter outros dados, como:
  - Requisitos não Funcionais relacionados
  - Diagrama de atividades relacionado
  - Protótipo de interface
  - Outros diagramas
  - ...
- ▶ O importante é que as necessidades sejam entendidas e acordadas por todas as partes interessadas!

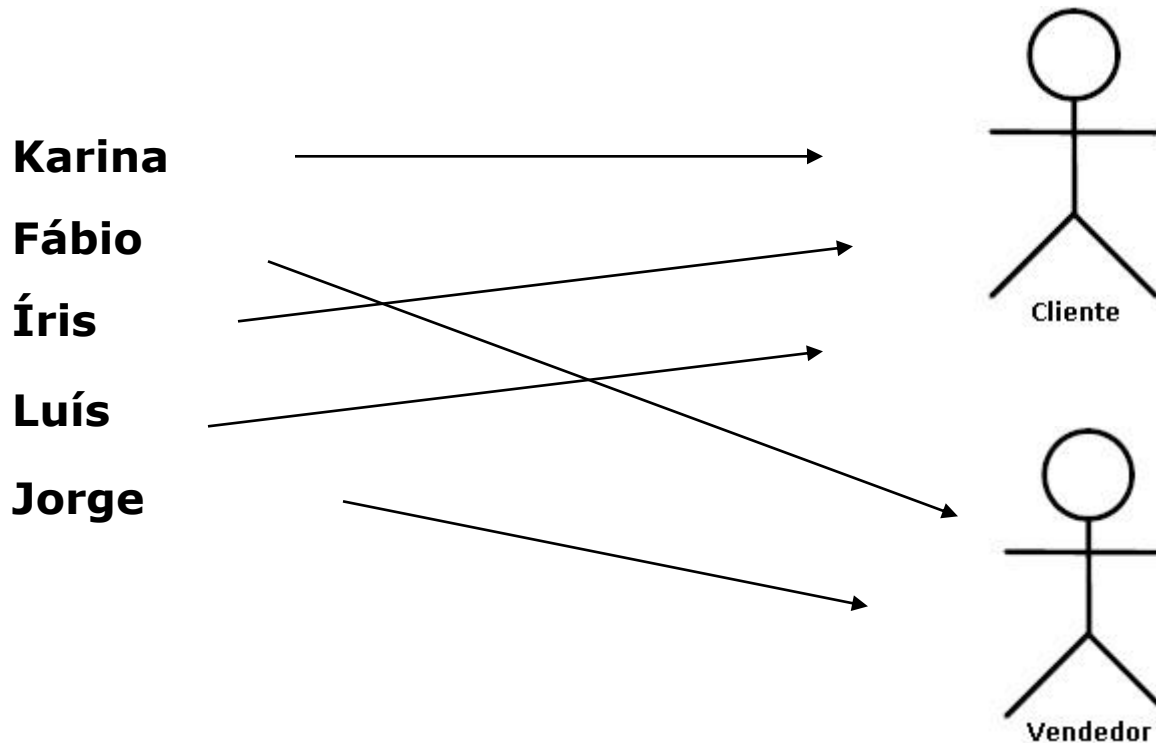
# Atores

- ▶ Casos de uso são executados por **atores**
- ▶ Eles constituem as entidades externas do ambiente do sistema
- ▶ São papéis que os usuários do sistema devem desempenhar nas interações
- ▶ Uma “instância de ator” pode ser desempenhada tanto por um indivíduo quanto por um sistema ou mesmo por um dispositivo



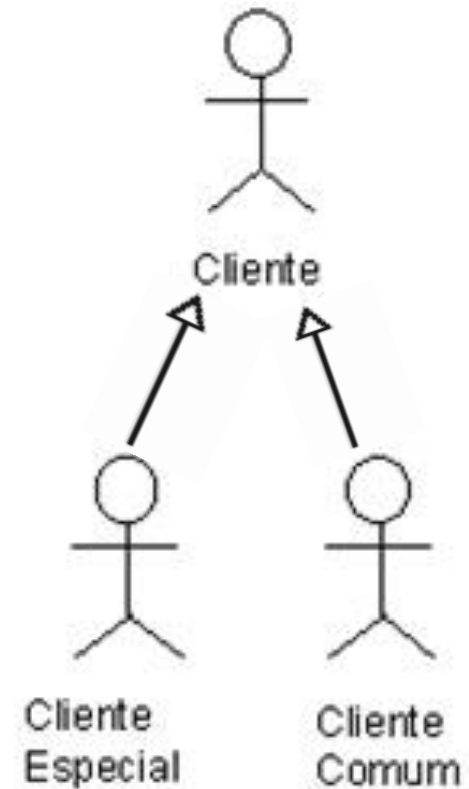
# Atores

- ▶ Lembre-se, Atores representam papéis/perfis e não pessoas



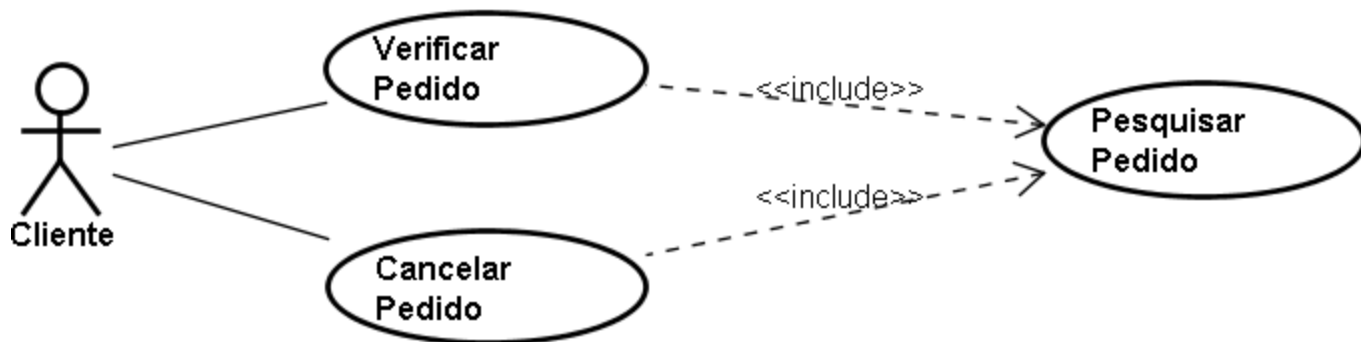
# Atores: Especialização

É possível definir tipos gerais de atores e especializá-los usando o relacionamento de especialização



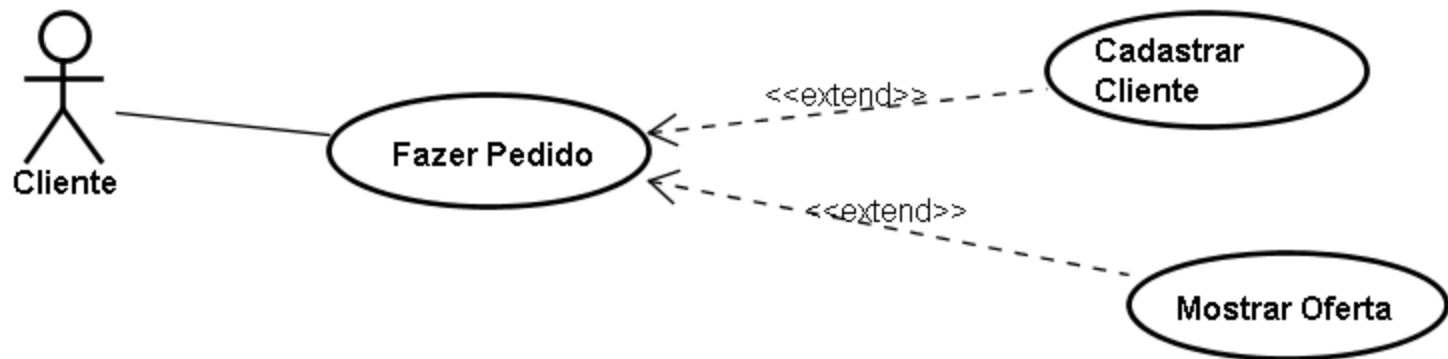
# Relacionamento: Inclusão

- ▶ Um Caso de Uso base incorpora o comportamento de outro Caso de Uso
- ▶ O relacionamento é utilizado para evitar a descrição do mesmo fluxo de eventos várias vezes



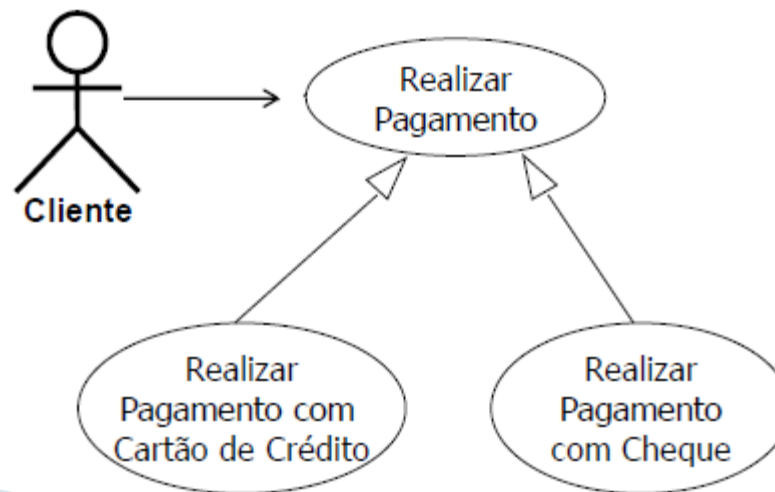
# Relacionamento: Extensão

- ▶ Modela partes opcionais da execução de um Caso de Uso
- ▶ Modela fluxos que são executados somente em determinados casos, sob determinadas circunstâncias ou que dependem de escolha de um ator



# Relacionamento: Generalização

- ▶ Relaciona um Caso de Uso especializado a um mais geral
- ▶ O filho herda o comportamento do pai, podendo adicionar e redefinir passos em pontos arbitrários do comportamento original



# Resumo dos Relacionamentos

## ▶ Inclusão

- Use quando o mesmo comportamento se repete em mais de um Caso de Uso e o processo de realizar X **sempre** envolve realizar Y pelo menos uma vez

## ▶ Extensão

- Use quando você quiser modelar um comportamento **opcional** de um Caso de Uso

# Resumo dos Relacionamentos

- ▶ Generalização entre Casos de Uso
  - Use quando você identificar Casos de Uso semelhantes e um deles for uma forma especial (uma especialização) do outro
- ▶ Generalização entre Atores
  - Use quando um ator (filho) é um **tipo de** outro ator mais genérico (pai)

# Tipos de Casos de Uso

- ▶ Concreto
  - É iniciado por um ator e constitui um fluxo completo de eventos
- ▶ Abstrato: nunca é instanciado diretamente
  - Casos de Uso abstratos geralmente são:
    - Incluídos em outros Casos de Uso
    - Estendidos de outros Casos de Uso
    - Generalizações de outros Casos de Uso
- ▶ Atores “enxergam” apenas casos de uso concretos

# Modelo de Casos de Uso

- ▶ É um modelo completo das funções do sistema em termos de Casos de Uso
- ▶ A finalidade mais importante é comunicar, de forma fácil de entender, o comportamento do sistema ao usuário final
- ▶ Contém:
  - Casos de uso, Atores, Relacionamentos
  - Pacotes de Caso de uso, Diagramas de Caso de Uso, Especificações, etc...

# Benefícios

- ▶ Casos de Uso são focados no usuário do sistema, assim as *reais* necessidades são tratadas logo cedo
- ▶ São fáceis de entender
- ▶ Facilitam o acordo entre todas as partes interessadas
- ▶ Pode ser usado no levantamento, elicitação e validação dos requisitos, conectando todas as etapas

# Exercícios [5]

**(SERPRO – CESPE 2010)**

[68] A descrição dos cenários de uso com informações acerca da utilização do sistema sob diversos pontos de vista e formas de operação deve fazer parte do levantamento dos requisitos.

**(BASA – CESPE 2007)**

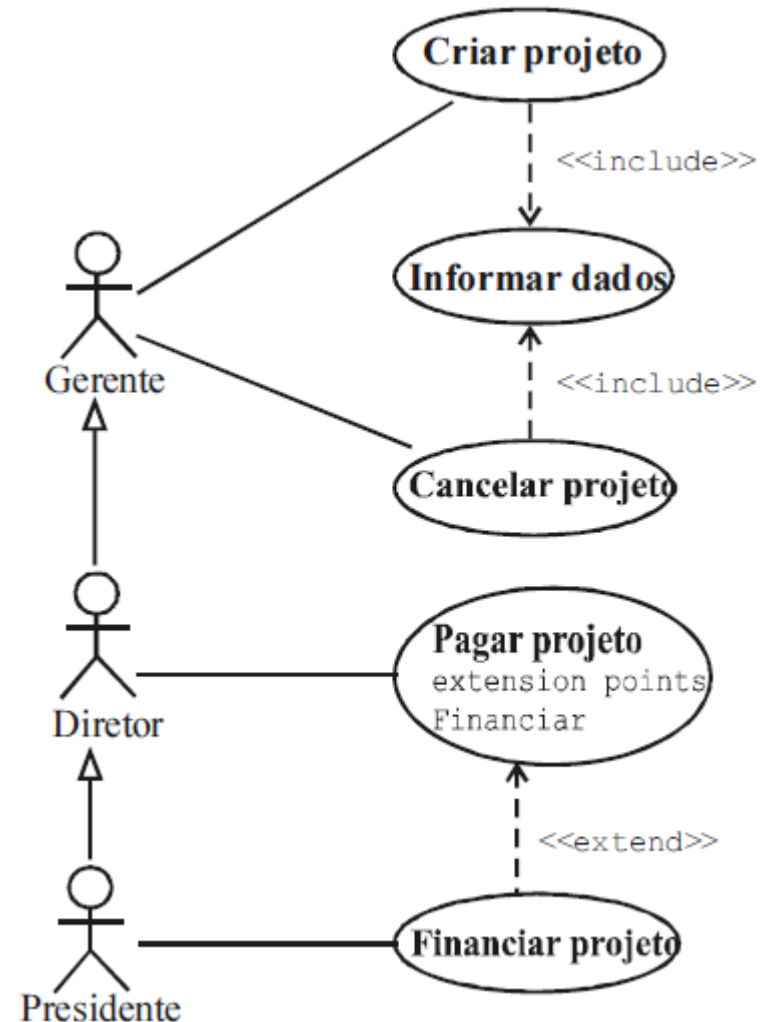
[66] A construção de um modelo de casos de uso é um meio para capturar requisitos funcionais com foco no valor dos requisitos para os usuários. Um caso de uso especifica uma seqüência de ações que o sistema pode realizar e que produzem resultados observáveis e de valor para os atores.

[67] Em um modelo de casos de uso, pode haver diferentes tipos de usuários representados por atores. Além de tipos de usuários, atores podem representar outros sistemas ou hardwares que interagem com o sistema a ser desenvolvido. Atores se comunicam com o sistema via casos de uso.

# Exercícios [5]

(MPE/RR – CESPE 2008)

[87] No diagrama UML ao lado, o ator Presidente está relacionado ao caso de uso Criar projeto; o caso de uso Informar dados contém comportamento comum a dois casos de uso; o caso de uso Pagar projeto estende o comportamento Financiar projeto e Cancelar projeto é abstrato.



# Análise de Requisitos

- ▶ Depois que os requisitos foram coletados, os produtos de trabalho servem como base para a análise de requisitos
- ▶ A análise de requisitos visa a descobrir alguns problemas e torná-los mais consistentes antes da especificação formal

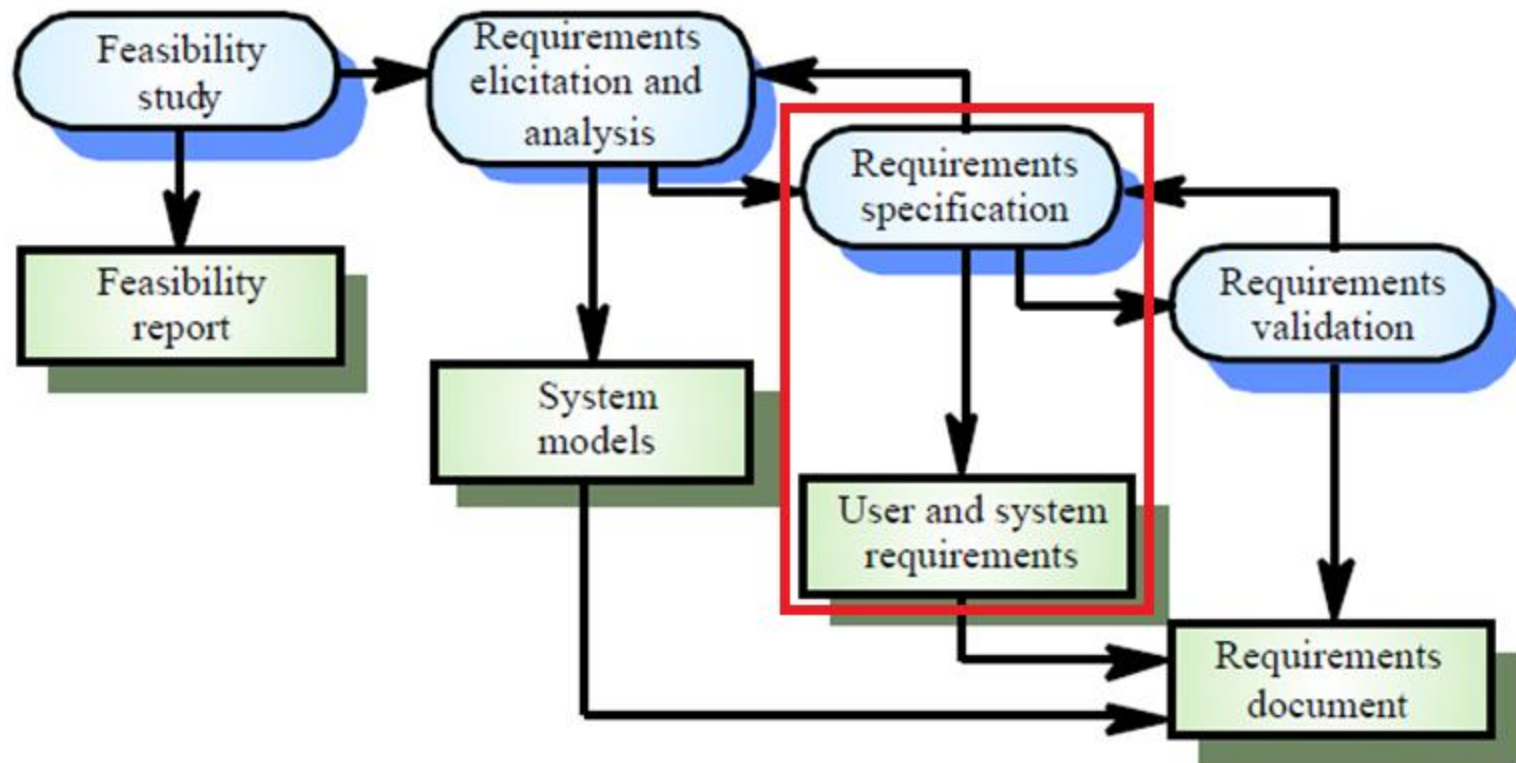
# Atividades

- ▶ Classificação e organização
  - Agrupar requisitos relacionados e os organizá-los em conjuntos coerentes
- ▶ Checagens de:
  - Consistência
  - Ambiguidade
  - Omissões
  - Relacionamentos entre requisitos, etc.
- ▶ Priorização e negociação
  - Priorizar requisitos e negociar conflitos

# Negociação

- ▶ A atividade de negociação é importante para que o analista possa conciliar os conflitos entre os *stakeholders*
  - Eles pedem mais do que pode ser feito
  - Ou têm “necessidades especiais”
- ▶ É papel do analista de requisitos balancear todas essas demandas
- ▶ Requer grande capacidade de interação social

# Especificação de Requisitos



# Especificação de Requisitos

- ▶ O termo especificação tem vários significados , podendo ser:
  - Um documento escrito
  - Um modelo gráfico
  - Um modelo matemático formal
  - Uma coleção de cenários de uso, etc.
- ▶ A abordagem utilizada depende da necessidade específica de cada projeto
  - Documentos escritos combinados com modelos gráficos para sistemas maiores
  - Cenários de uso para sistemas mais simples, etc.

# Especificação do Sistema

- ▶ É o produto final produzido pelo engenheiro de requisitos
- ▶ Serve como a base para
  - Engenharia de Software
  - Engenharia de Hardware
  - Engenharia de Banco Dados, etc.
- ▶ Descreve a função de um sistema de software e as restrições impostas a ele
- ▶ Também descreve as informações que entram e saem do sistema

# Exercícios [6]

(SERPRO – CESPE 2010)

[95] O documento de requisitos de software estabelece formalmente o que os desenvolvedores de sistema devem implementar e inclui a especificação resumida dos requisitos do sistema e a visão detalhada da arquitetura do sistema.

(MPOG – ESAF 2008)

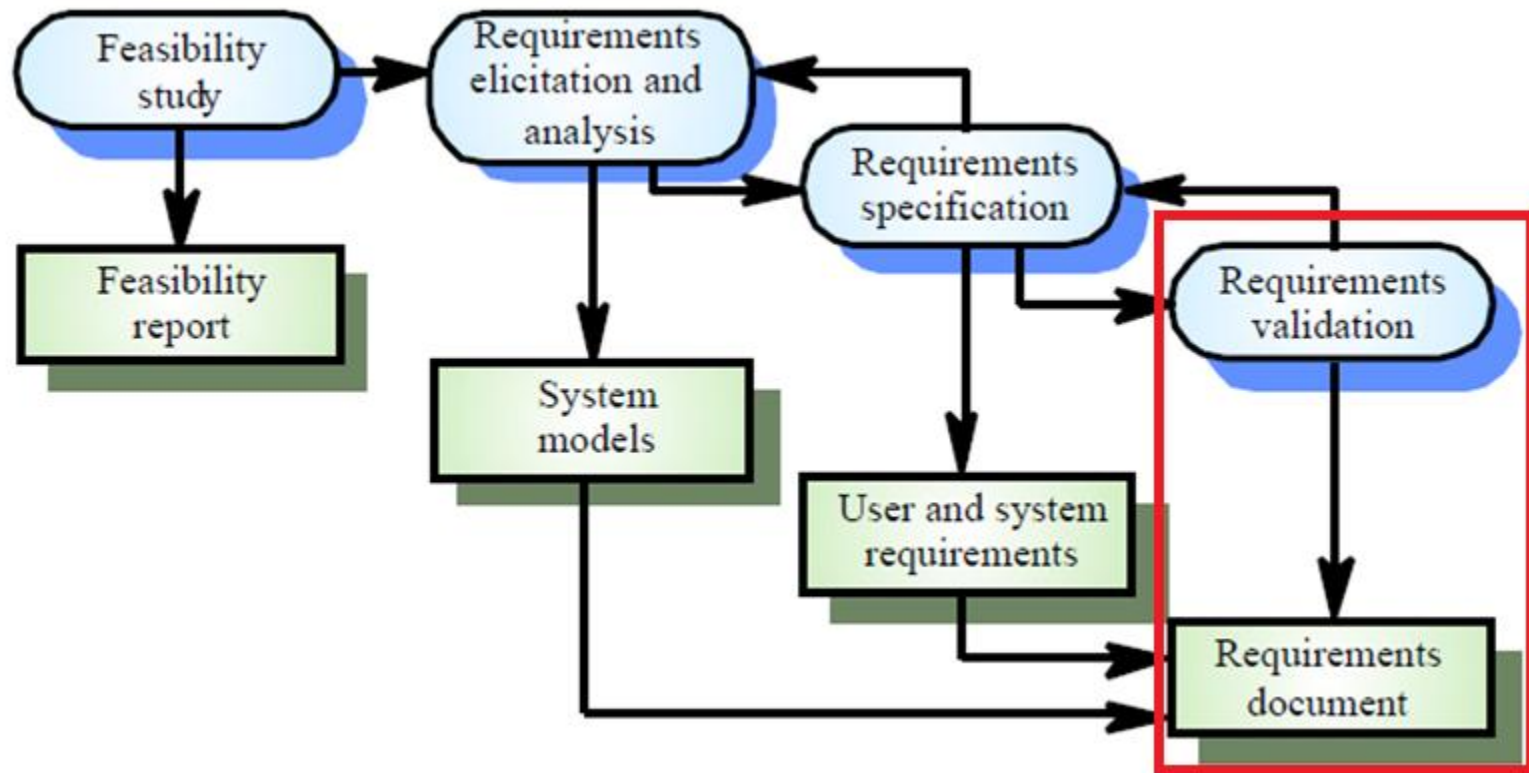
[16-e] Avaliar se os requisitos associados ao desempenho, ao comportamento e às características operacionais do sistema foram explicitamente declaradas é uma tarefa de especificação de requisitos.

# Exercícios [6]

(ANATEL – CESPE 2009)

[76] A elicitação de requisitos ocorre usualmente antes da fase de análise de requisitos, e resulta na produção de uma especificação precisa das necessidades do usuário bem como dos requisitos do sistema a ser desenvolvido, o que exige maior interação social por parte do responsável pela elicitação, quando relacionada à exigência de interação durante a fase de análise.

# Validação de Requisitos



# Objetivos

- ▶ Demonstrar que os requisitos definem o sistema que o usuário realmente deseja
- ▶ Visa a assegurar que
  - A versão do documento de requisitos descreve as funcionalidades e características do sistema satisfatoriamente
  - Os requisitos são consistentes e de alta qualidade
  - O documento de requisitos provê uma base adequada para Projeto e Implementação

# Entradas e Saídas

## Entradas

- ▶ O documento de requisitos (preliminar)
- ▶ Padrões organizacionais
- ▶ Conhecimento implícito da organização

## Saídas

- ▶ Lista de problemas com os requisitos
- ▶ Ações acordadas para tratar destes problemas
- ▶ Documento de Requisitos aprovado (final)

# Principais técnicas

- ▶ Revisões (inspeções)
  - Um grupo de pessoas se reúne, lê e analisa os requisitos, para identificar problemas e suas possíveis soluções
- ▶ Prototipagem
  - Um protótipo executável demonstra os requisitos e ajudam os *stakeholders* a descobrir problemas
- ▶ Geração de Casos de Teste
  - Casos de teste ajudam a mostrar se os requisitos estão ambíguos ou incompletos

# Exercícios [7]

**(TRE/MT – CESPE 2010)**

[33-e] Revisão de requisitos, prototipação e geração de casos de teste são exemplos de técnicas de validação de requisitos.

**(MPE/AM – CESPE 2008)**

[58] Para validar os requisitos de um sistema, é melhor realizar apenas uma revista técnica formal no final da especificação, pois assim todos os requisitos são analisados de uma única vez.

**(MPE/AM – CESPE 2008)**

[56] Uma das formas de resolução de ambigüidades de requisitos consiste em realizar a prototipação de partes do sistema, antes de se adotar uma solução.

# Gerenciamento de Requisitos

# Gerenciamento de Requisitos

- ▶ É o processo de gerenciar as mudanças nos requisitos durante o processo de Engenharia de Requisitos
- ▶ Requisitos são, inevitavelmente, incompletos e inconsistentes
  - Novos requisitos surgem à medida que as necessidades de negócios mudam e há um melhor entendimento do sistema
  - Diferentes pontos de vista normalmente têm requisitos diferentes (e conflitantes)

# Requisitos sempre mudam!

- ▶ A prioridade de cada requisito muda ao longo do projeto
- ▶ Cliente não é a mesma coisa que Usuário
  - Perspectivas diferentes
- ▶ O ambiente de negócios e tecnológico do projeto muda durante o seu desenrolar
- ▶ É necessário gerenciar tudo isso

# Rastreabilidade

- ▶ Relacionam os requisitos e avaliam seus impactos
- ▶ Rastreabilidade de **Fonte**
  - Ligação entre o requisito e o *stakeholder* que o propôs (e sua necessidade original)
- ▶ Rastreabilidade de **Requisitos**
  - Ligações entre requisitos que dependem entre si
- ▶ Rastreabilidade de **Projeto**
  - Ligação entre o requisito e o projeto (arquitetura, módulos, código) do software

# Ferramentas

- ▶ É impossível rastrear requisitos sem uma ferramenta CASE adequada
- ▶ Ela deve:
  - Armazenar os requisitos em um ambiente seguro e gerenciado
  - Dar suporte ao gerenciamento de mudança dos requisitos
  - Permitir recuperar automaticamente a ligação (rastreabilidade) dos requisitos

# Exercícios [8]

**(IJSN – CESPE 2010)**

[64] A rastreabilidade de requisitos é essencial para que o controle de mudanças possa avaliar o impacto de uma solicitação de Mudança.

**(TRE/MT – CESPE 2010)**

[33-d] A matriz de rastreabilidade não oferece suporte para requisitos funcionais.

**(SERPRO – CESPE 2008)**

[92] A gerência de requisitos tem como objetivo principal controlar a evolução dos requisitos, seja por constatação de novas necessidades, seja por constatação de deficiências nos requisitos registrados até o momento. Um exemplo de gerência de requisitos é a aplicação de revisão por pares, que constata deficiências nos requisitos especificados.

# Exercícios [8]

(Governo do ES – CESPE 2009)

[72] O gerenciamento de requisitos deve compreender e controlar mudanças nos requisitos de sistema, além de avaliar os seus impactos. Para atingir esse propósito, podem ser mantidas informações de rastreabilidade a serem usadas para avaliar quais outros requisitos seriam afetados por uma mudança, bem como o impacto da mudança de requisitos no projeto e na implementação do sistema.

# Gabaritos dos Exercícios

- ▶ [1] – [35–a] E, [78] C, [79] E, [65] C
- ▶ [2] – [32] E
- ▶ [3] – [31–a] E, [33–a] E, [33–b] E, [33–c] E, [12] B
- ▶ [4] – [66] E, [51] C, [71] E
- ▶ [5] – [68] C, [66] C, [67] C, [87] E
- ▶ [6] – [95] E, [16–e] E, [76] E
- ▶ [7] – [33–e] C, [58] E, [56] C
- ▶ [8] – [64] C, [33–d] E, [92] E, [72] C

# FIM