



# Especificação de Requisitos

RALPH J. R. FILHO

# Conteúdo

- ▶ O que é e para quê serve um requisito?
- ▶ Porquê requisitos devem ser gerenciados?
- ▶ Como coletar requisitos?
- ▶ Como documentar requisitos?
- ▶ Como colocar em prática?
- ▶ Visualizar exemplos e praticar com exercícios.
- ▶ Compartilhar experiências.

# Referências

- ▶ Rational - "Applying Requirements Management with Use Cases" (artigo)
- ▶ Rational - "Traceability Strategies for Managing Requirements with Use Cases" (artigo)
- ▶ IBM - OpenUp Wiki ([epf.eclipse.org/wikis/openup/](http://epf.eclipse.org/wikis/openup/))
- ▶ "Best Practices for Agile/Lean Documentation" ([www.agilemodeling.com/essays](http://www.agilemodeling.com/essays))

# Chaos Report

Project Success Factors	% of Responses
1. User Involvement	15.9%
2. Executive Management Support	13.9%
3. Clear Statement of Requirements	13.0%
4. Proper Planning	9.6%
5. Realistic Expectations	8.2%
6. Smaller Project Milestones	7.7%
7. Competent Staff	7.2%
8. Ownership	5.3%
9. Clear Vision & Objectives	2.9%
10. Hard-Working, Focused Staff	2.4%
Other	13.9%

Project Challenged Factors	% of Responses
1. Lack of User Input	12.8%
2. Incomplete Requirements & Specifications	12.3%
3. Changing Requirements & Specifications	11.8%
4. Lack of Executive Support	7.5%
5. Technology Incompetence	7.0%
6. Lack of Resources	6.4%
7. Unrealistic Expectations	5.9%
8. Unclear Objectives	5.3%
9. Unrealistic Time Frames	4.3%
10. New Technology	3.7%
Other	23.0%

# Cenário

- ▶ Os clientes fazem solicitações.
- ▶ As solicitações são implementadas e testadas no software.
- ▶ Como documentar estas solicitações?
- ▶ Quais são os modelos (templates) mais adequados?
- ▶ Como manter documentação com pouco tempo disponível?
- ▶ Aonde manter a documentação e quem deve fazê-lo ?
- ▶ Como estabelecer uma ligação entre as atividades de desenvolvimento e testes com as requisições?
- ▶ Como saber em que versão cada solicitação foi entregue?
- ▶ Como podemos rastrear mudanças?

# O que é um requisito?

- ▶ "Uma condição ou capacidade com a qual o sistema deve estar em conformidade" (Rational / IBM)
- ▶ "Uma capacidade do software necessária para resolver um problema de um usuário ou cumprir determinado objetivo" (IEEE)



# O que é gestão de requisitos?

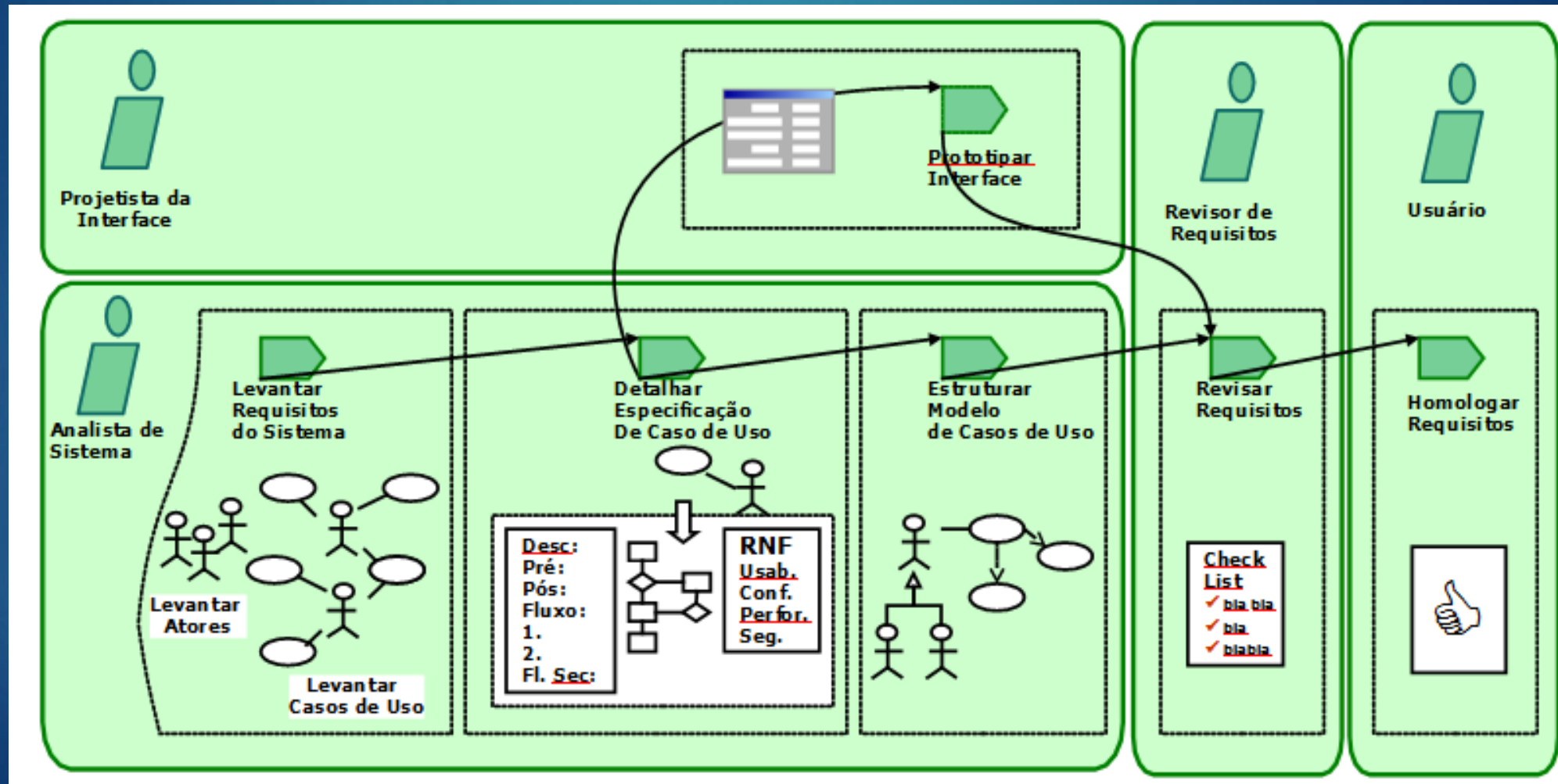
- ▶ "Uma abordagem sistemática para elicitar, organizar e documentar os requisitos do sistema"
- ▶ "Um processo que estabelece e mantém um acordo entre o cliente e o time do projeto durante a mudança dos requisitos do sistema" (Rational / IBM)

# Desafios

- ▶ Requisitos não são sempre óbvios e possuem muitas fontes
- ▶ Requisitos não são fáceis de descrever de forma clara
- ▶ Existem diversos tipos de requisitos em diferentes níveis de detalhes
- ▶ O número de requisitos pode ficar ingerenciável se não for controlado
- ▶ Requisitos devem ser rastreáveis
- ▶ Requisitos possuem propriedades únicas
- ▶ Os requisitos mudam com frequência



# Atividades Desempenhadas

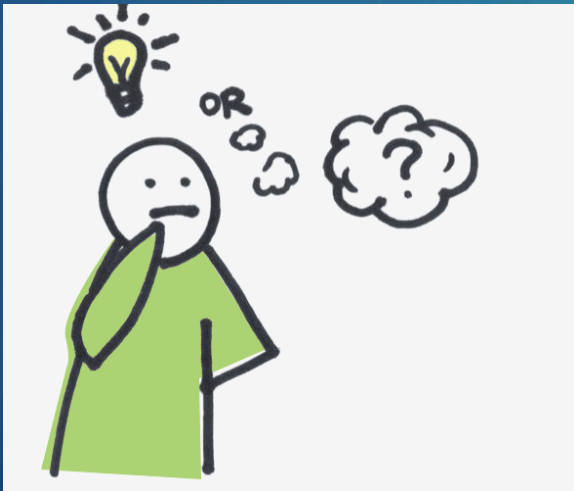


# Artefatos

- ▶ Necessidades de usuário
- ▶ Funcionalidades do Produto
- ▶ Requisitos (funcionais e não-funcionais)
- ▶ Regras de Negócio
- ▶ Itens de Glossário
- ▶ Atores
- ▶ Casos de Uso
- ▶ Protótipos de Interface

# Necessidade de Usuário

- Representa uma função desempenhada por usuário que deve ser mapeada. Pode ser descrito como "Estória".



# Exemplo de Necessidade

## 1. Processo de Compras

- ▶ Em uma empresa de comércio varejista, o usuário responsável pelo processo de compras, aqui denominado "comprador", necessita enviar listas de produtos aos seus fornecedores para que estes possam preencher cotações de preços. As cotações de preços são comparadas pelo comprador para que o mesmo selecione os produtos de cada fornecedor naquilo que representa maior vantagem para a empresa que fará a compra. Um produto pode ser mais vantajoso quando possuir uma combinação dos fatores: menor preço, menor prazo de entrega e menor incidência de impostos. Depois de selecionados, os produtos são organizados em pedidos de compra que o comprador envia por e-mail para os fornecedores.

# Funcionalidade do Produto

- Representa uma característica funcional do produto, ou um conjunto de ações desempenhadas pelo sistema que atende uma necessidade de usuário





# Funcionalidade do Produto

## ▶ **Exemplos**

- ▶ Registro de usuário e tela de boas-vindas
- ▶ Cadastro e controle de grupos de produtos
- ▶ Emissão de notas fiscais
- ▶ Emissão de relatórios de contas a receber com diferentes visões



# Requisito Funcional

- ▶ São características do sistema que transformam-se em implementações. Devem ser testáveis. Exemplos:
- ▶ **Auditoria.** Deve rastrear quem fez uma ação no sistema e quando fez?
- ▶ **Autenticação.** O acesso ao sistema será controlado?
- ▶ **Licenciamento.** Como será a aquisição e o monitoramento de licenças?
- ▶ **Impressão.** Deve imprimir arquivos? Como procederá com impressões?
- ▶ **Relatórios.** O sistema irá gerar relatórios? Como irá proceder?
- ▶ **Agendamento.** Algumas ações poderão ser agendadas?
- ▶ **Segurança.** Como será o esquema de acesso a partes do sistema?

# Requisito Funcional

## ► Exemplos

- O software deve permitir que o usuário modifique a ordem das colunas de cada listagem, salvando esta preferência para acesso futuro.
- O software deve permitir que o usuário marque telas específicas do sistema como favoritas, oferecendo um atalho para que o usuário retorne nesta tela rapidamente.
- O software deve garantir que todo usuário possua uma senha com nível de complexidade aceitável utilizando um algoritmo para esta verificação. Além disso, o usuário deve trocar a sua senha a cada 12 meses.

# Requisito Não-Funcional

- ▶ São atributos ou qualidades do sistema. Devem ser testáveis.
- ▶ **Usabilidade**. Documentação (help), facilidade de aprendizado.
- ▶ **Confiabilidade**. Políticas de tolerância a falhas, habilidade de recuperação de falhas.
- ▶ **Desempenho**. Tempo de resposta de processamento, uso de recursos.
- ▶ **Segurança**. Privacidade, integridade.
- ▶ **Distribuição**. Política de versionamento, entrega aos clientes.
- ▶ **Padronização**. Interfaces gráficas, componentes.
- ▶ **Restrições**. Hardware, software (plataformas).

# Requisito Não-Funcional

## ► Exemplos

- O sistema deve executar tarefas de armazenamento de dados retornando em até 10 segundos após o comando do usuário. Caso a operação ainda não tiver sido completada, deve ser executada em segundo plano.
- O sistema, para ser executado, necessita do Sistema Operacional Windows versão XP ou 7, 32 bits, com mínimo de 1024 MB de RAM e processador Dual Core de 2.1 GHz de frequência.
- O sistema deve providenciar uma forma de restrição de acesso aos usuários à telas específicas, mostrando mensagem de erro caso ocorra uma tentativa de acesso não-autorizado.

# Regra de Negócio

- ▶ São políticas com as quais o sistema deve entrar em conformidade.
- ▶ Podem conter leis e regulamentos impostos ao negócio.
- ▶ Devem ser descritas de maneira tão clara que pode ser transformada em código-fonte.
- ▶ Algumas regras de negócio podem ser aplicáveis a mais de um caso de uso. Estas regras devem ser centralizadas.
- ▶ Exemplo
- ▶ O número de componentes da equipe deve ser menor ou igual a dez.
  - ▶ `team.getMembers() <= 10;`



# Regra de Negócio

## ► Exemplos

- Se um pedido é cancelado e se o pedido não tiver sido enviado, então o pedido deve ser fechado.
- O pedido deve ser enviado ao cliente apenas se o cliente tiver endereço de entrega cadastrado.
- Um pedido refere-se a um produto, no mínimo.
- Um cliente deve ser considerado "bom" se todas as faturas enviadas a ele que não foram pagas tem menos do que 30 dias.
- O preço líquido de um produto é igual a:  $\text{custo do produto} * (1 + \text{porcentagem total de imposto} / 100)$



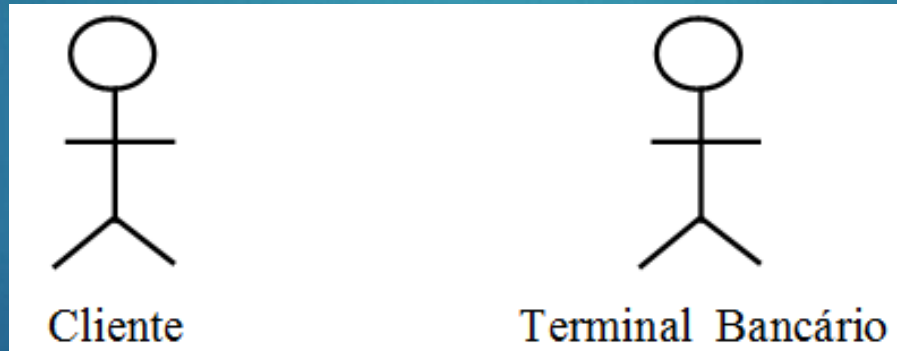
# Item de Glossário

- ▶ Define termos importantes usados no projeto.
- ▶ É importante para garantir o entendimento dos conceitos de negócio de forma uniforme tanto pelo cliente quanto pela equipe.
- ▶ Inclui termos, definições, sinônimos e referências

GLOSSÁRIO	
✓ <b>AMINOÁCIDO</b>	Molécula básica ("tijolo") a construção de proteínas
✓ <b>BASES NITROGENADAS</b>	Substâncias cujas moléculas, aos pares, formam os "degraus" da dupla hélice do DNA
✓ <b>CÓDON</b>	Grupo de três bases nitrogenadas na cadeia de DNA que especifica um aminoácido na síntese de uma determinada proteína
✓ <b>DNA</b>	Abreviação em inglês de "Ácido DesoxirriboNucléico", a substância cuja molécula em forma de dupla hélice é o principal componente dos cromossomos
✓ <b>EXPRESSION</b>	"Leitura" (transcrição em RNA) de uma sequência de DNA e sua tradução numa sequência de aminoácidos (síntese da proteína)
✓ <b>GENE</b>	Termo cunhado em 1909, designava uma unidade hereditária abstrata. Com a biologia molecular, passou a ser a sequência de DNA que especifica os aminoácidos de uma proteína
✓ <b>NUCLEOTÍDEO</b>	Unidade básica do DNA, composta por uma única base ("letra")
✓ <b>PROTEÍNA</b>	Tipo de molécula complexa que está na base do metabolismo e da estrutura celulares, composta por longas cadeias de aminoácidos, cuja sequência determina sua estrutura e sua função
✓ <b>RNA</b>	Abreviação em inglês de "Ácido RiboNucléico", substância aparentada com o DNA, do qual se diferencia por formar normalmente uma fita simples e pela base uracila (U) no lugar de timina (T)

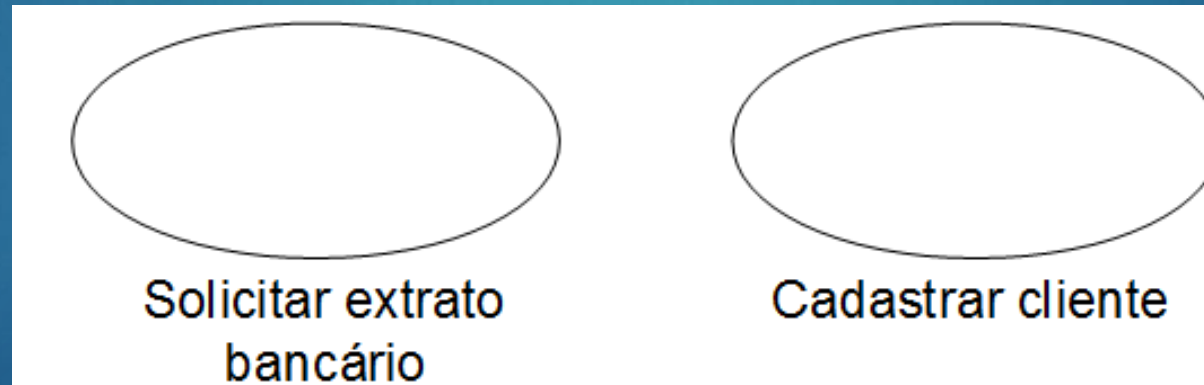
# Ator

- ▶ São pessoas ou outros sistemas que interagem com o sistema em questão.
- ▶ O termo "ator" é padrão UML e a representação gráfica é como na imagem abaixo.

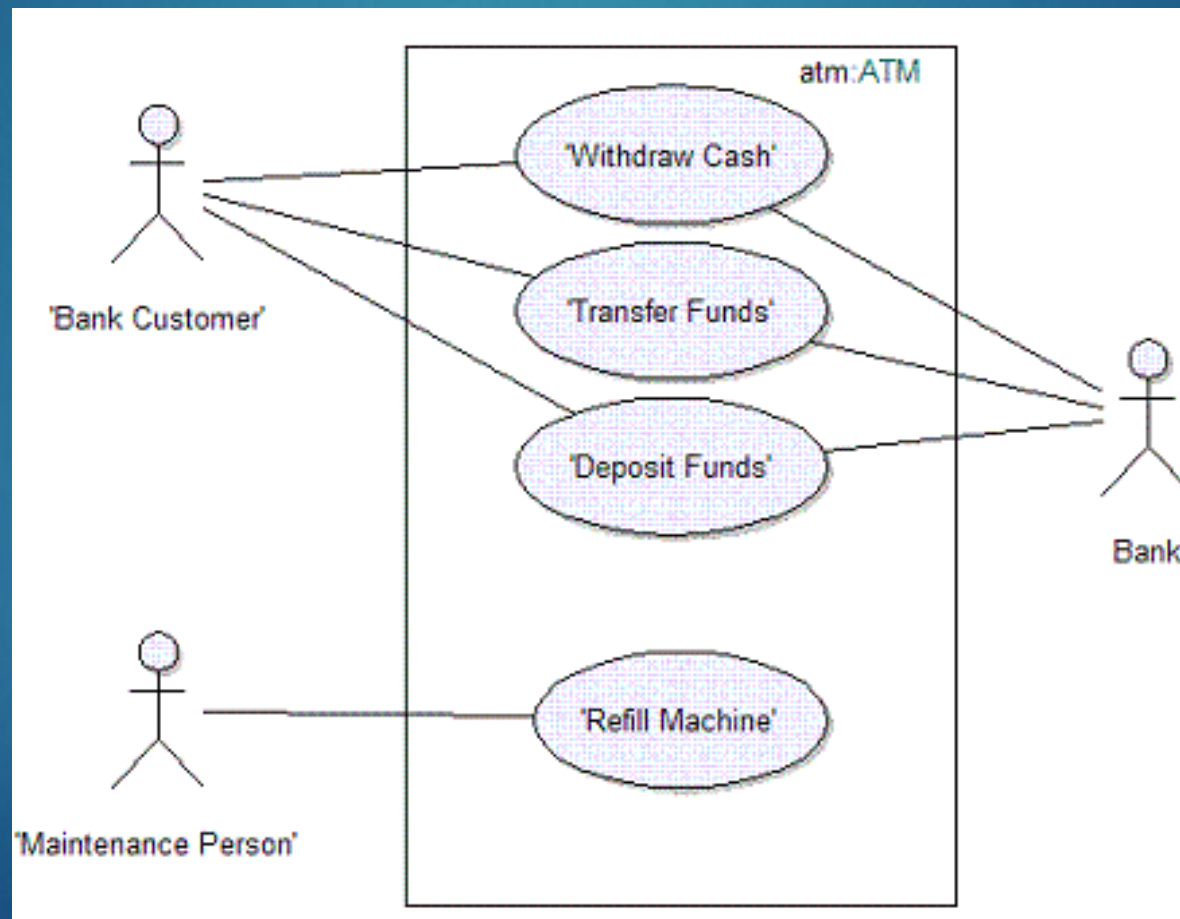


# Caso de Uso

- ▶ Descreve um conjunto de ações que o usuário faz ou que o sistema executa de forma automática.
- ▶ O termo "caso de uso" é padrão UML e a representação gráfica é feita conforme a figura abaixo.



# Diagrama de Caso de Uso



# Exemplo

## **Caso de Uso 01 – Cadastrando Cliente (descrição típica)**

---

**Ator Primário:** Cliente

**Precondições:** Nenhuma

### **Fluxo Normal**

- 1 – Cliente preenche ficha cadastral,
- 2 – Assistente de Cadastro informa recebimento documentação cadastral
- 3 – Gerente de Cadastro informa aprovação de Cliente

**Fluxo Alternativo:** documentação incompleta ou com erro

- 2a – Assistente de Cadastro Informa documentação irregular.
  - 2b – Cliente envia documentação corrigida para cadastro.
- Retorna ao passo 2.

**Fluxo Alternativo:** irregularidade nos dados cadastrais

- 3a – Gerente de Cadastro informa irregularidade dados cadastrais
- 3b – Cliente atualiza dados cadastrais.
- 3c - Retorna ao passo 3.



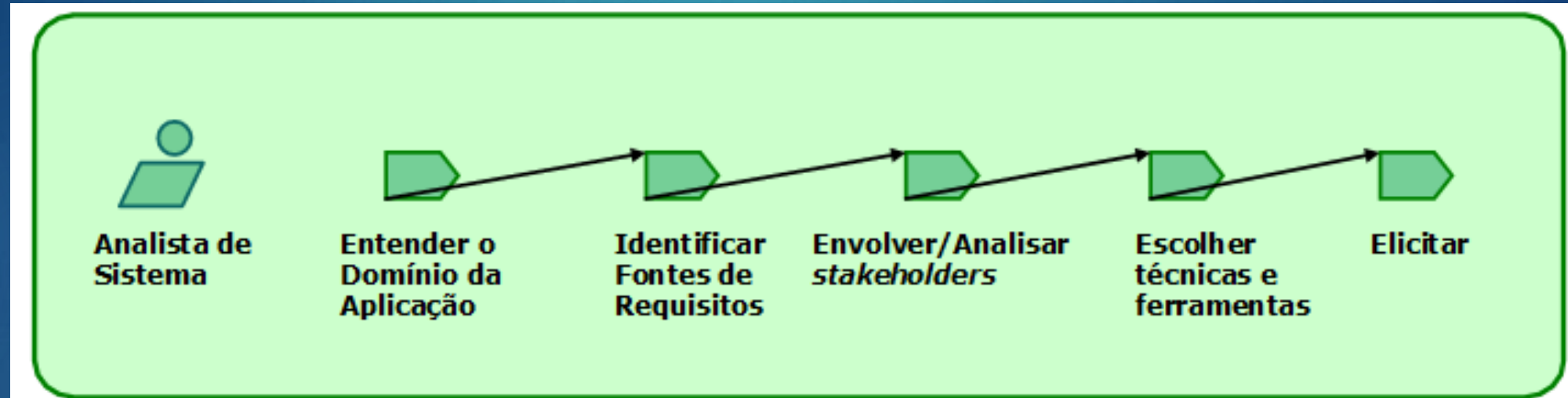
# Protótipo de Interface

- ▶ Define um modelo que será utilizado para implementar uma "tela" do sistema.
- ▶ Evidentemente, apenas aplicável se a funcionalidade envolve "telas".





# Processo de Elicitação



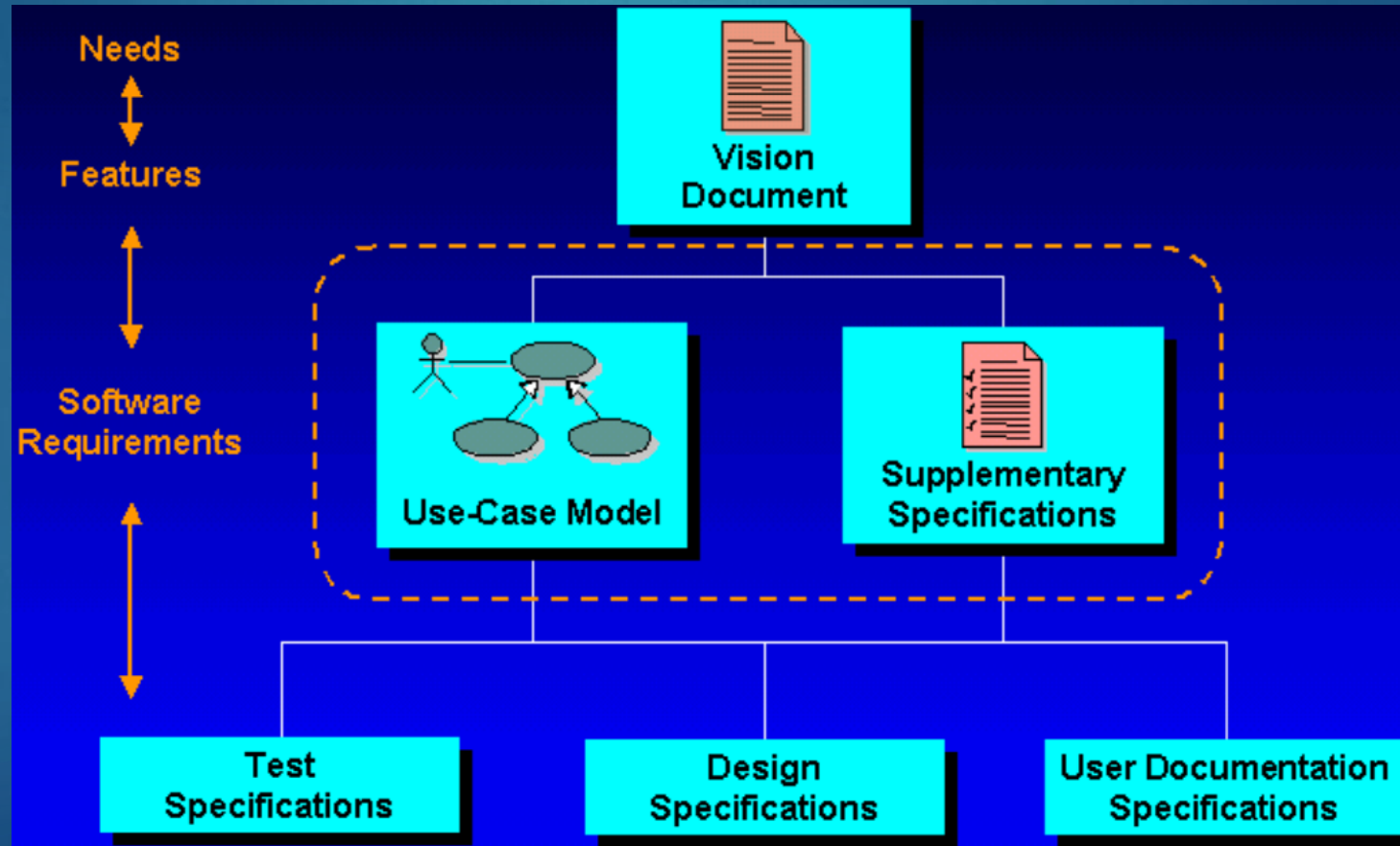
# Processo de Elicitação

- ▶ **Entender o domínio da aplicação.** Investigar detalhes do processo do cliente descrevendo os processos atuais ("as is") e propor mudanças ("to be").
- ▶ **Identificar fontes de requisitos.** Stakeholders, especialistas, documentação existente, relatórios, manuais, etc.
- ▶ **Envolver/Analisar Stakeholders.** Fazer com que os interessados participem do projeto em todo seu ciclo de vida.
- ▶ **Escolher técnicas e ferramentas.** Definir técnicas de abordagem dos usuários.

# Técnicas de Elicitação

- ▶ **Entrevista.** Pode ter lista de tópicos ou ser uma conversa informal.
- ▶ **Questionário.** Preparado previamente pode ser enviado para os usuários.
- ▶ **Análise de Domínio.** Estudar documentação e software existentes.
- ▶ **Introspecção.** Basear-se no que o analista acredita que o cliente precisa.
- ▶ **Brainstorming.** Reunião com stakeholders para gerar ideias.
- ▶ **Etnografia.** Observação dos usuários durante suas atividades habituais.
- ▶ **Aprendizagem.** O analista é treinado por um usuário experiente como se fosse desempenhar um papel dentro do cliente.

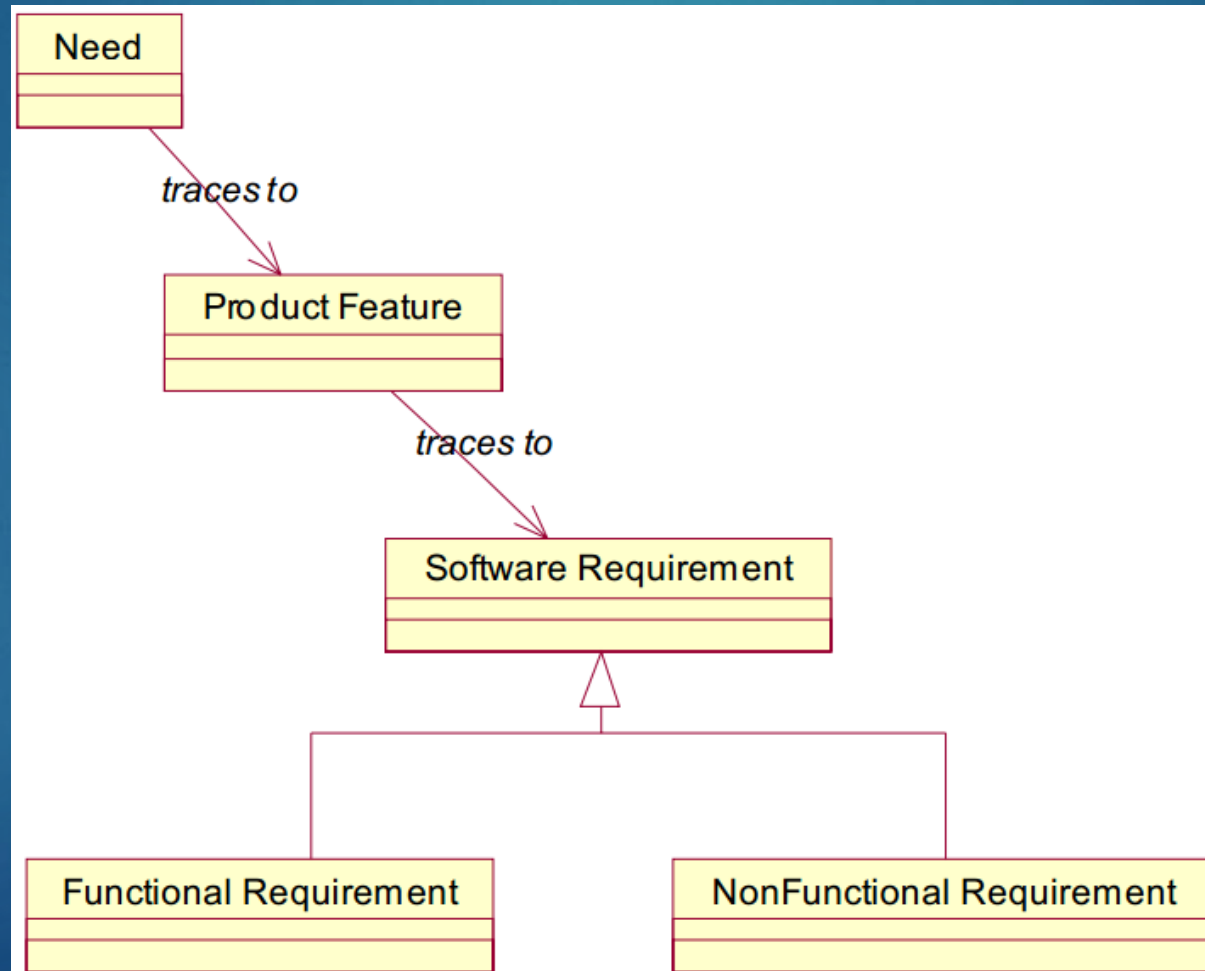
# RUP - Fluxo de Solicitações



# Estratégias

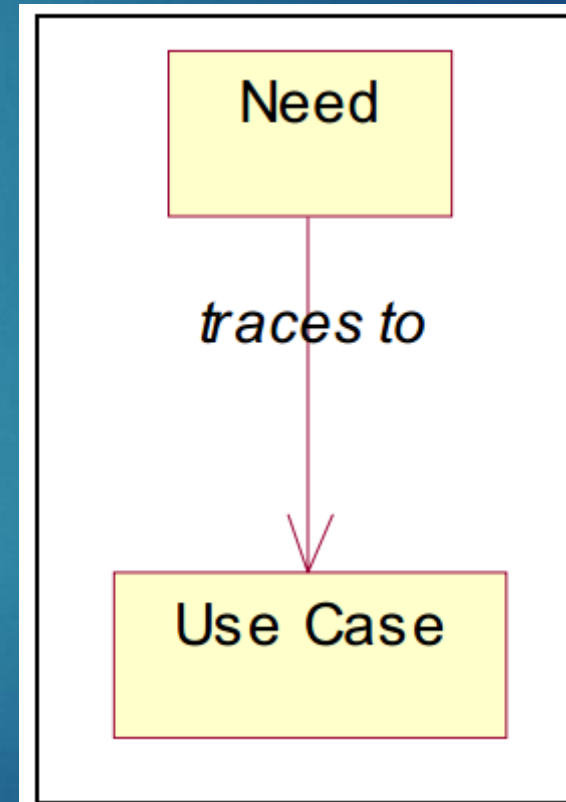
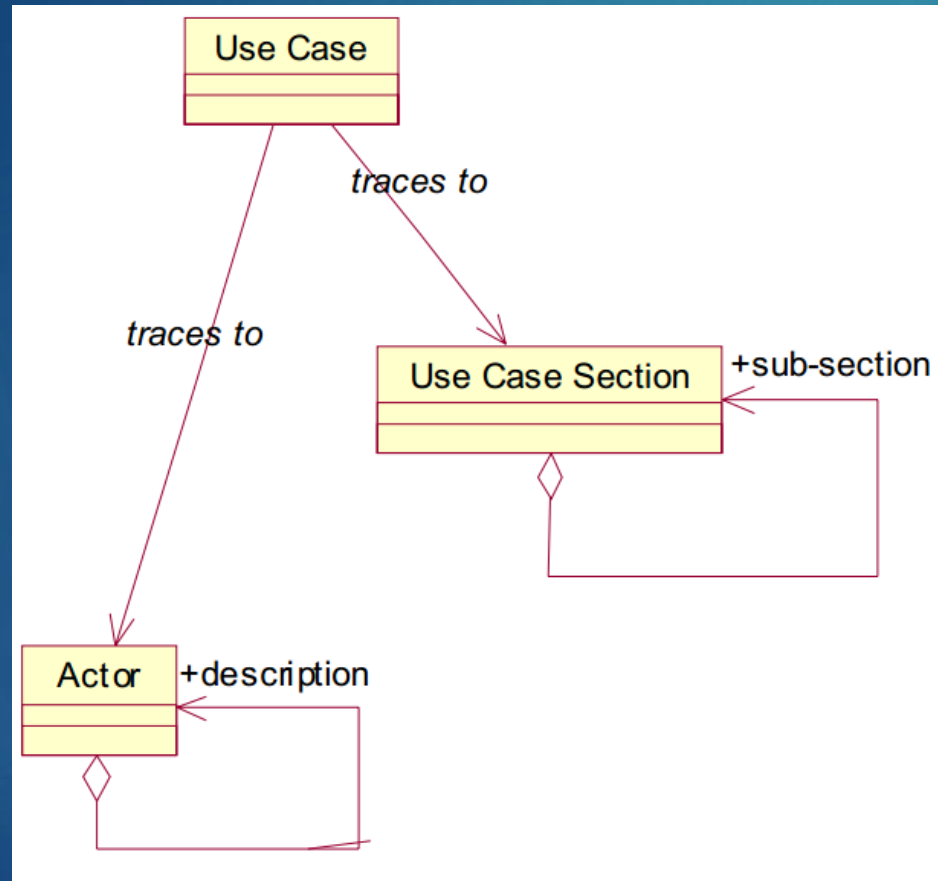
1. Modelo sem caso de uso
2. Apenas modelo de Caso de Uso
3. Funcionalidades norteadas pelo modelo de Caso de Uso (RUP)
4. O modelo de Caso de Uso é uma interpretação da Especificação de Requisitos do Software (ERS / SRS)

# Modelo sem casos de uso

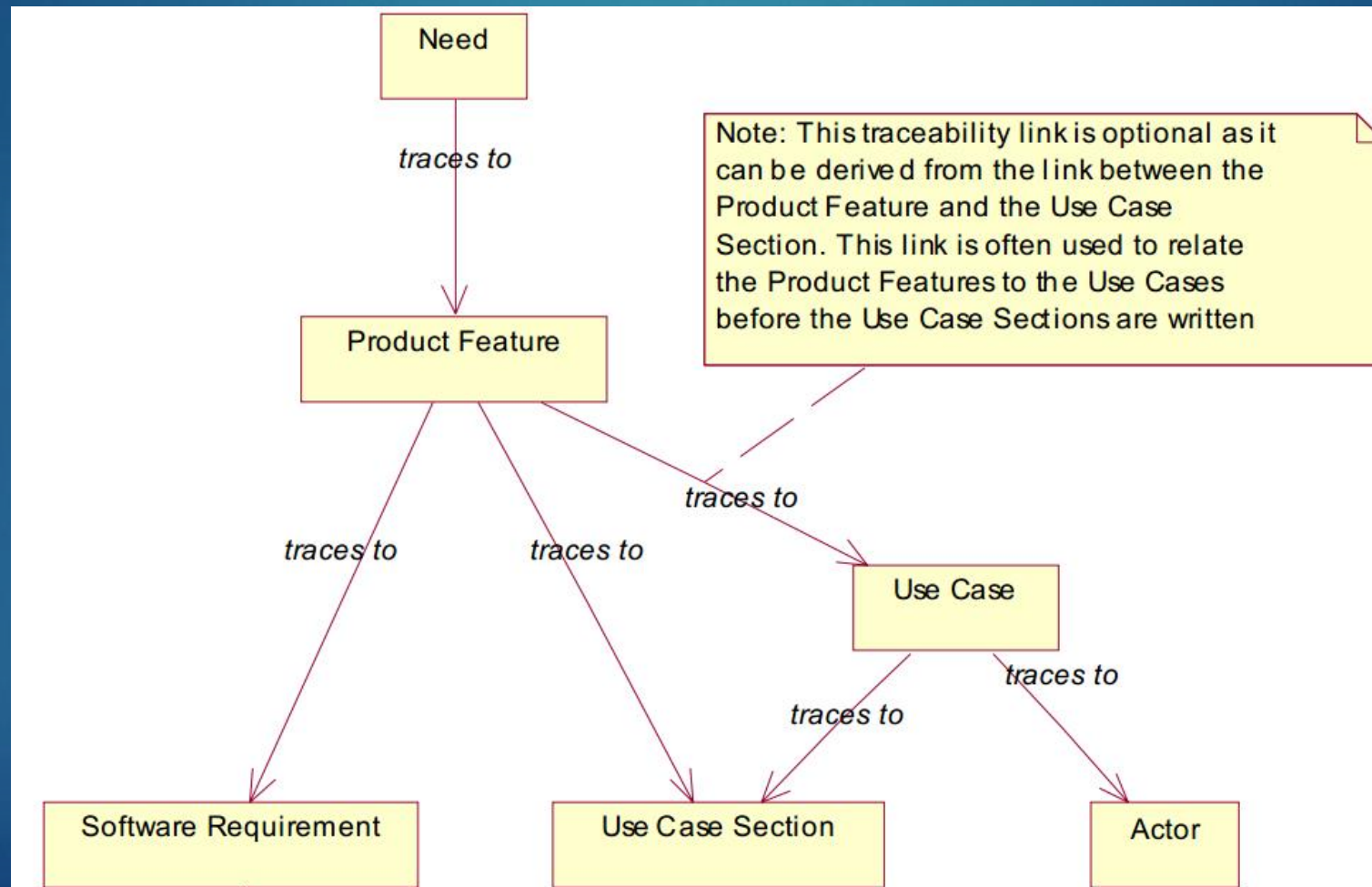




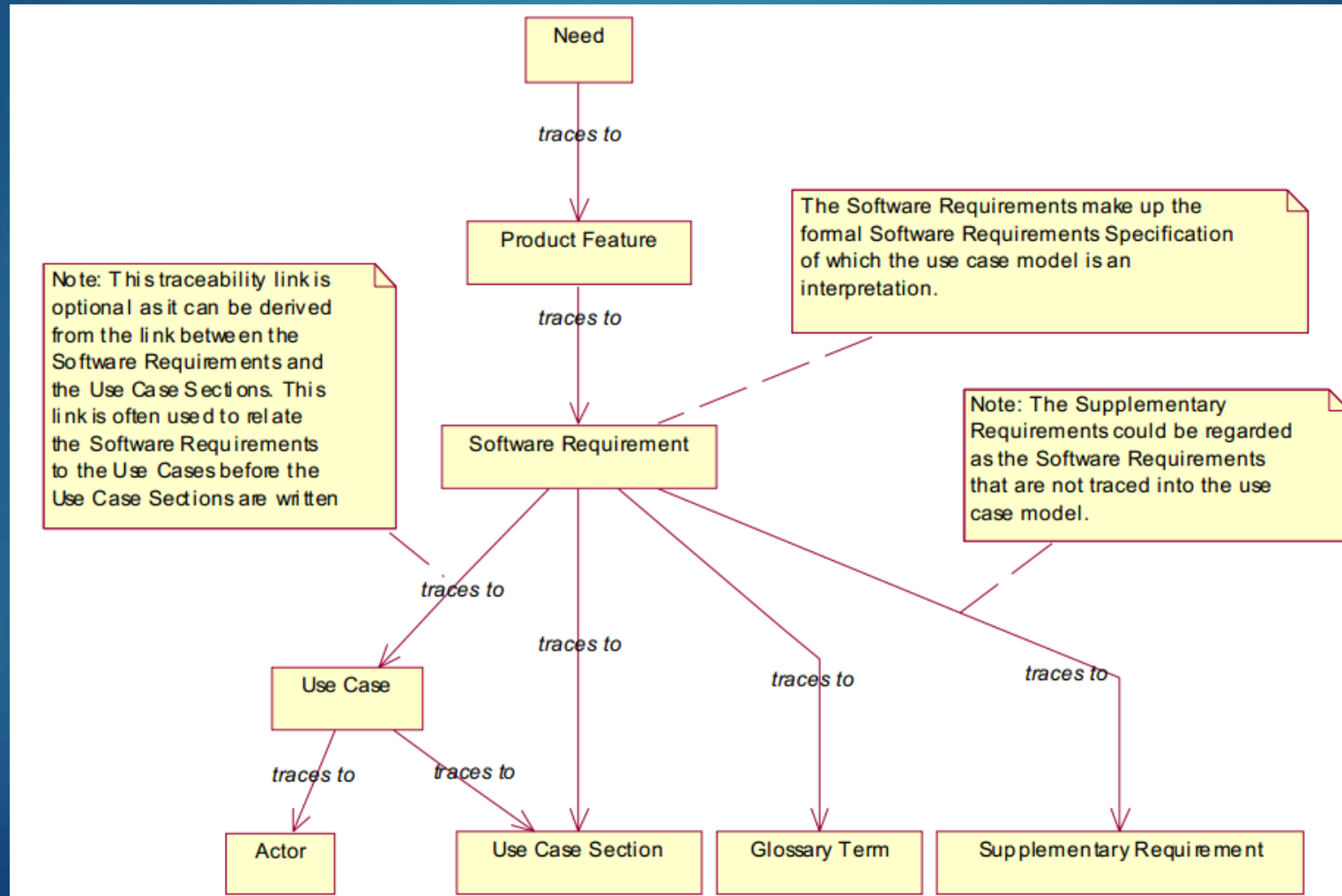
# Apenas modelo de Caso de uso



# Funcionalidades norteiam o modelo de Caso de Uso (RUP)





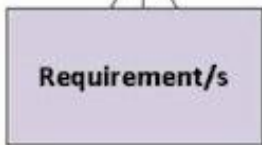
# O modelo de Caso de Uso é uma interpretação da SRS



# Práticas ágeis

- ▶ Prefira especificação executável ao invés de documentos estáticos.
- ▶ Documente conceitos estáveis e não ideias especuladas.
- ▶ Documente o mais tarde possível.
- ▶ Gere a documentação a partir dos fontes.
- ▶ Mantenha a documentação simples e clara.
- ▶ Mantenha poucos documentos e sem sobreposição.
- ▶ Coloque as informações em local apropriado e público.
- ▶ Documente com propósitos claros e objetivos.
- ▶ Foque na necessidade do usuário do documento.
- ▶ Dedique-se a uma escrita eficiente.

# Práticas ágeis (news.dice.com)

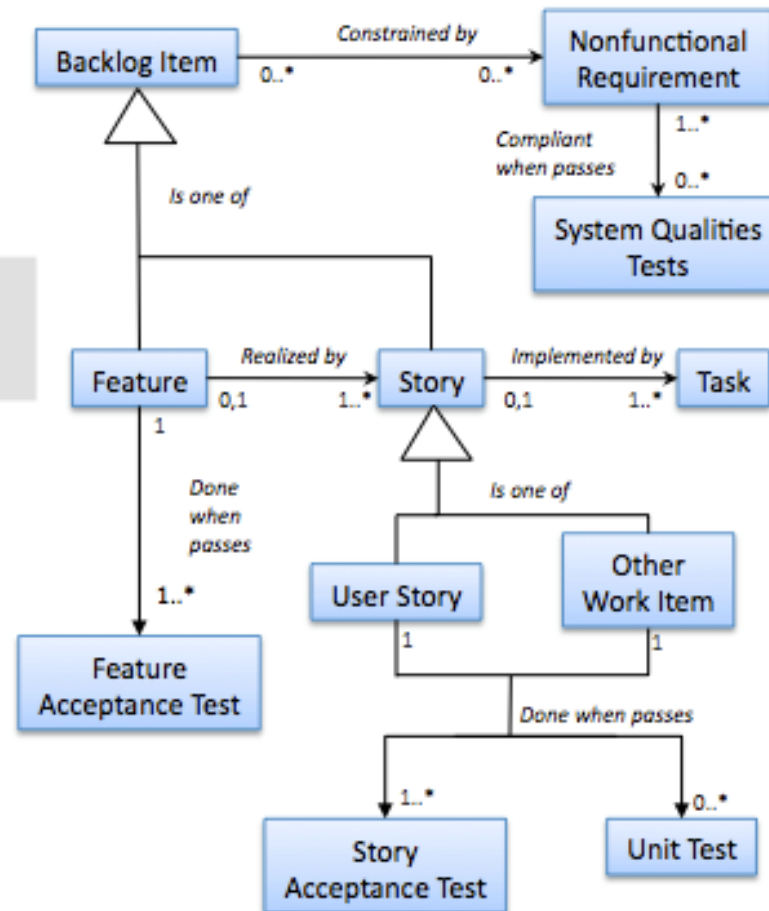
Complexity	Artifact	Information Captured	Participants	Lifecycle Approaches
Simple <i>Conceptual Level</i>		<ul style="list-style-type: none"><li>• Users</li><li>• Scenario</li><li>• Expectation</li></ul>	<ul style="list-style-type: none"><li>• End User</li><li>• SME</li><li>• BA</li></ul>	Agile & Waterfall
More Complex <i>Functional Requirement</i>		<ul style="list-style-type: none"><li>• Users</li><li>• Preconditions</li><li>• Behaviors / Events</li><li>• Alternative Paths</li><li>• Failure-conditions</li><li>• Post-conditions</li></ul>	<ul style="list-style-type: none"><li>• End User</li><li>• SME</li><li>• BA</li><li>• Architect</li><li>• Lead Developer</li></ul>	
Detailed <i>Technical Requirement</i>		<ul style="list-style-type: none"><li>• Description (include reference to User Story / Scenario and Use Case)</li><li>• Events / Process Flow</li><li>• Data Definitions</li><li>• System Information</li><li>• Performance</li><li>• Test Case</li><li>• Failure-recovery</li><li>• Post-conditions</li></ul>	<ul style="list-style-type: none"><li>• SME</li><li>• Architect</li><li>• Lead Developer</li></ul>	Waterfall

Copyright 2012 Semantech Inc.



# Práticas ágeis

Source: *Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise*.  
Copyright 2010, Leffingwell, LLC.  
See [www.scalingsoftwareagility.wordpress.com](http://www.scalingsoftwareagility.wordpress.com)



# Boas práticas

- ▶ "O faturista deve ser capaz de emitir 10 notas fiscais em menos de 02 horas".
- ▶ Este requisito tem um sujeito (ator), um estado mensurável e final (10 notas fiscais) e um critério de performance (02 horas).
- ▶ Defina um requisito por vez.
  - ▶ "O piloto deve poder controlar o ângulo da aeronave usando uma mão".
  - ▶ "O piloto deve poder sentir o ângulo de inclinação através do controle de inclinações da aeronave".

# Boas práticas

- ▶ Evite conjunções (e, ou) que englobam vários requisitos.
  - ▶ "O navegador deve poder ver a posição da aeronave relativo à rota demonstrada pelo radar".
  - ▶ "O navegador deve poder ver a posição da aeronave conforme estimado pelo guia de inércia".
- ▶ Evite palavras que implicam em exceções (a menos que, exceto se, se necessário, mas, porém).
  - ▶ "O design deve providenciar assentos voltados para a parte traseira da aeronave para cada membro da tripulação".
- ▶ Utilize sentenças simples e diretas.
  - ▶ "O piloto deve ser capaz de ver o indicador de velocidade".

# Boas práticas

- ▶ Utilize linguagem natural.
- ▶ "Para fazer uma ligação de longa distância, o usuário deve tirar o telefone do gancho. O sistema deve responder com um tom de discagem. O usuário deve digitar um "9". O sistema deve responder com um tom de discagem distinto [...]"
- ▶ "O sistema consiste em quatro estados: Ocupado, Tom de Discagem, Tom de Discagem Distinto e Conectado. Para mudar do status de Ocupado para o status de Tom de Discagem, tire o telefone do gancho. Para mudar do status de Tom de Discagem para o Tom de Discagem Distinto, digite um "9"."
- ▶ Qual dos dois textos é mais claro para o usuário e para a equipe?

# Desafios da Elicitação

## Common Requirements Problems

#1 Can't track changes 71%

#2 Difficult to write 70%

#3 Feature creep 67%

#4 Not well organized 54%



# Desafios

- ▶ Para detalhar os requisitos o analista precisa enfrentar uma série de desafios.
- ▶ Elicitar é tornar explícito, obter o máximo de informações possíveis para ter o domínio do negócio.
- ▶ Os usuários podem não ter uma ideia exata do que necessitam.
- ▶ Os usuários tem dificuldade para descrever seu conhecimento sobre um problema.
- ▶ Os usuários tem pontos de vista diferentes dos analistas.
- ▶ Os usuários podem dificultar o processo de análise se não forem favoráveis ao novo sistema.

# Desafios

- ▶ Requisitos devem atender a necessidade do cliente agregando valor ao mesmo. Deve-se evitar a prática de "gold plating".
- ▶ Requisitos devem ser consistentes e completos sem contradições.
- ▶ Requisitos devem ser viáveis dentro do orçamento e prazo disponíveis.
- ▶ Requisitos devem ser rastreáveis entre si. Importante para gerenciar mudanças (inevitáveis) e a evolução do requisito.
- ▶ Requisitos devem ser passíveis de priorização. Um bom exemplo é a técnica EID (Essencial, Importante e Desejável).

# Considerações Finais

- ▶ Os processos de gestão de requisitos precisam ser adaptados e adequados de acordo com a realidade da Empresa, a Natureza dos projetos, o tempo de dedicação para documentação e a experiência dos envolvidos.
- ▶ Quais artefatos serão gerados e gerenciados, deve, portanto, ser uma decisão da equipe de desenvolvimento.
- ▶ Boas fontes de estudos são: RUP / OpenUp, SCRUM, Livros de Engenharia de Software.
- ▶ Os modelos de processos ou frameworks não são doutrinas ortodoxas. São sim, guias contendo boas práticas.

# Considerações Finais

- ▶ Escolha com sabedoria.
- ▶ Se documentar de menos, pode comprometer o entendimento do sistema e fazer com que aspectos importantes do negócio sejam esquecidas ou fiquem na cabeça de quem elicitou / desenvolveu / testou.
- ▶ Se documentar demais, pode comprometer recursos humanos valiosos e gerar documentos extensos e confusos que, com o passar do tempo, serão abandonados. Deixando de ser atualizados ficam defasados e tornam-se inúteis.