



# UML 2.0 - Modelo Casos de Uso

*Márcia Ito*

[ito@mind-tech.com.br](mailto:ito@mind-tech.com.br)

*Julho/2004*

# Pensamento Inicial

"Nada lhe posso dar que já não exista em você mesmo. Não posso abrir-lhe outro mundo de imagens, além daquele que há em sua própria alma. Nada posso lhe dar a não ser a oportunidade, o impulso, a chave. Eu o ajudarei a tornar visível o seu próprio mundo, e isso é tudo."

(Hermann Hesse)

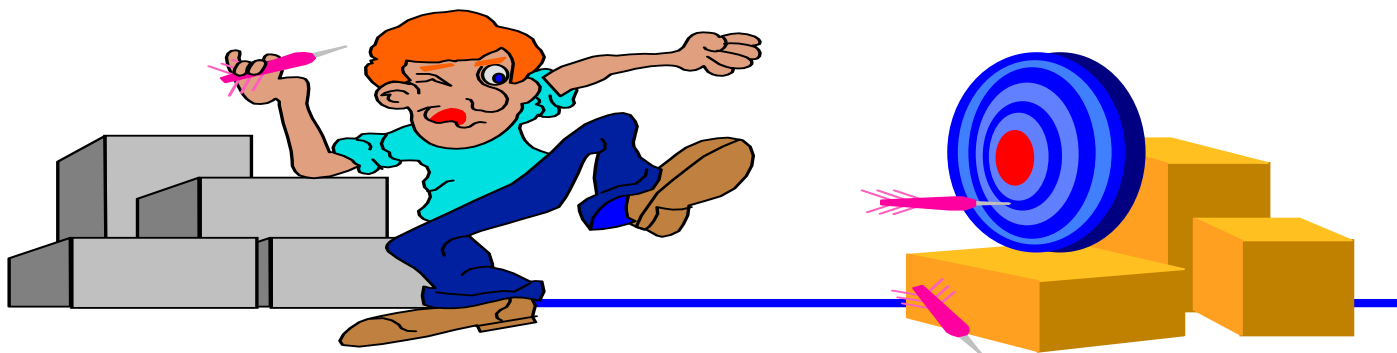
# Informações Gerais

“É impossível para um homem aprender aquilo que ele acha que já sabe.”

Epíteto

# Objetivos

- Entender a importância da análise do problema e do conhecimento das necessidades dos stakeholders e usuários antes de iniciar o desenvolvimento de software;
- Conhecer o processo de definição do sistema a partir das necessidades do usuário e do stakeholder;
- Aprender a extrair da definição do sistema com casos de uso, utilizando a notação da UML 2.0



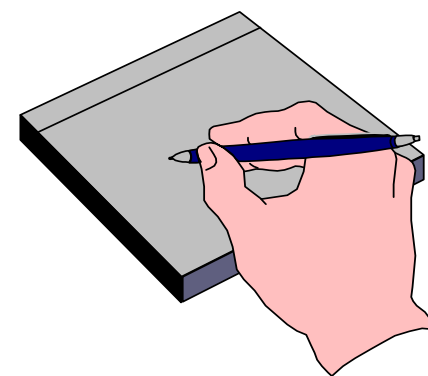
# Pré-Requisitos

- Desejável experiência em desenvolvimento de sistema.



# Informações Úteis

- Horários
- Material Didático
- Coffee-Break
- Telefones e recados
- Celulares e Pagers
- Estacionamento



# Introdução

“Não basta possuir um intelecto vigoroso; o primeiro requisito é aplicá-lo corretamente.”

René Descartes

# Problemas no Desenvolvimento de Software

Sintomas	Causas	Melhores Práticas
Necessidade não atendidas	Requisitos insuficientes	Desenvolvimento iterativo
Requisitos expirados	Comunicação ambígua	Gerenciar requisitos
Módulos não se integram	Arquitetura frágil	Arquitetura componetizada
Difícil manutenção	Complexidade absurda	Verificação contínua da qualidade
Descoberta tardia de falhas	Inconsistências não detectadas	Gerenciar mudanças
Baixa qualidade	Testes pobres	Modelagem visual (UML)
Baixa performance	Avaliação subjetiva	
Colisão de desenvolvedores	Desenvolvimento em "Cascata"	
Build-and-realease	Mudanças não controladas	

# UP – Unified Process

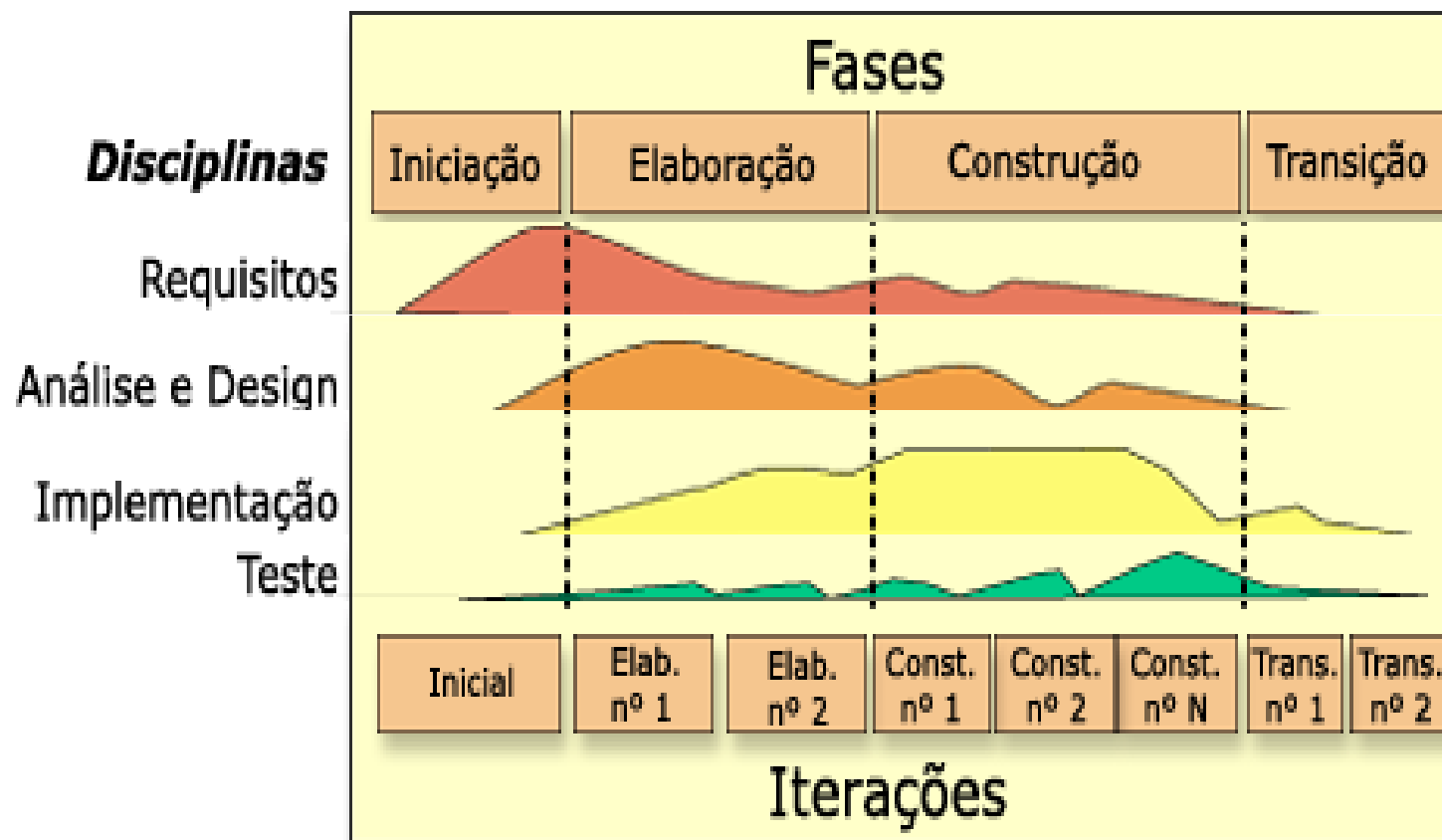
“Deve-se ter por certo que não há nada mais difícil de realizar, nem de êxito mais duvidoso, nem mais perigoso de empreender, do que dar início a uma nova ordem das coisas.”

Nicolau Maquiavel

# Objetivo

- Permitir o desenvolvimento de sistemas de alta qualidade
  - usabilidade
  - custos
  - prazos
- Utilizar as melhores práticas no desenvolvimento de sistemas
- Permitir o gerenciamento adequado na organização do desenvolvimento

# Modelo UP

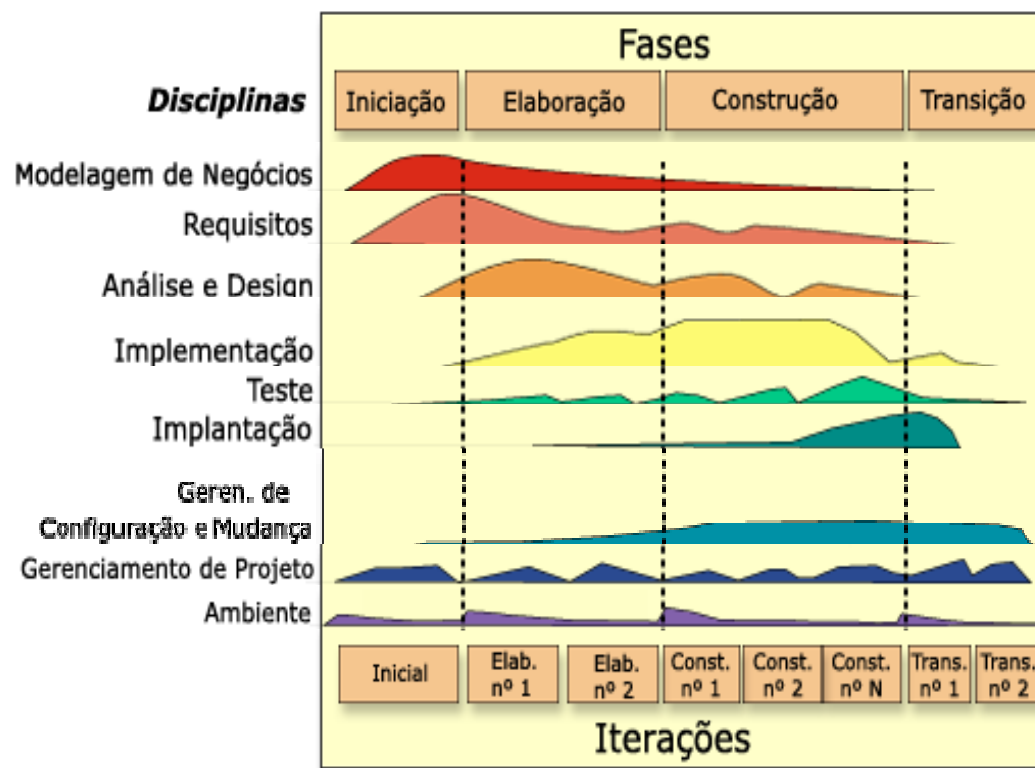
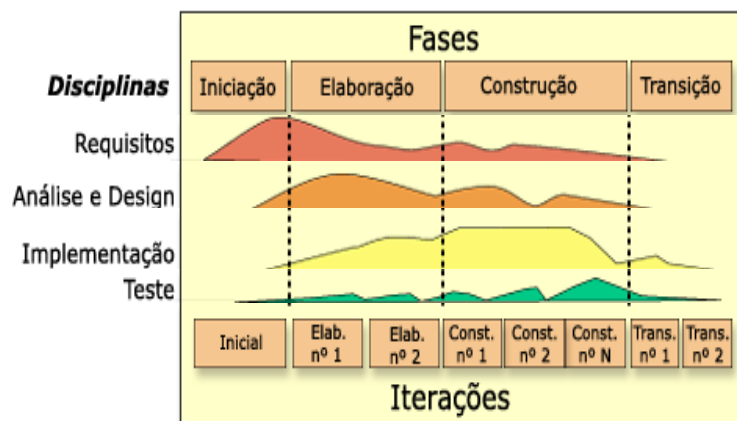


# Rational Unified Process - RUP®

- O Rational Unified Process é uma implementação refinada do UP, comercialmente disponível na forma de um conjunto de páginas em HTML.
- Ela foi elaborada para utilizar como ferramenta de apoio a Suite da Rational.



# UP x RUP<sup>®</sup>



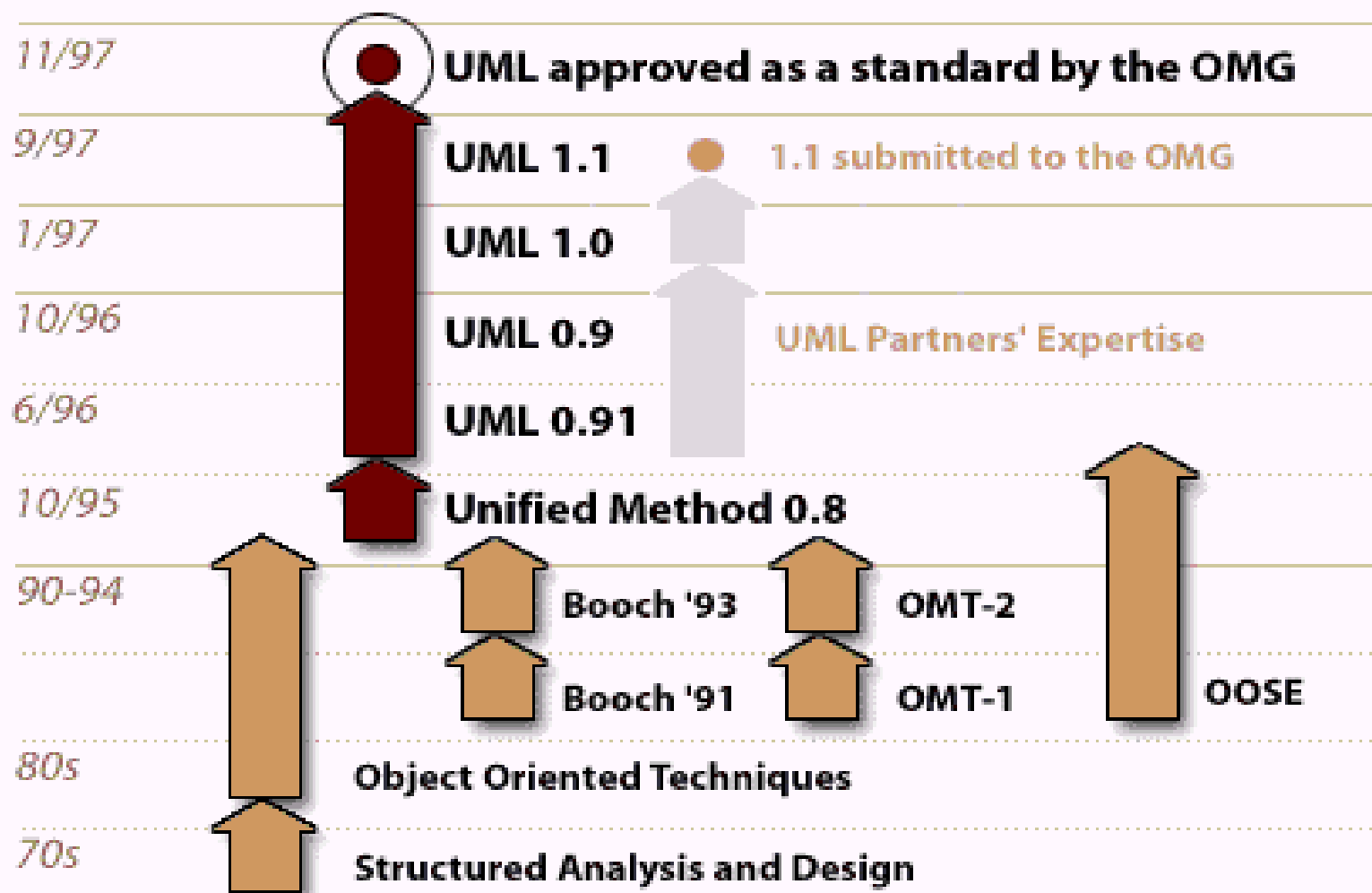
# Unified Modeling Language (UML)

“Não deixe extinguir-se a sua  
inspiração e a sua imaginação; não se  
torne um escravo do seu modelo.”

Vicent Van Gogh

# Evolução da UML

## Evolution of the Unified Modeling Language



# O que é OMG-UML?

- É uma linguagem que pode ser utilizada para especificar, visualizar, construir e documentar sistemas, através de modelos.
- É não proprietária e aberta a todos.
- Representa uma coleção de práticas de engenharia que comprovadamente se demonstraram eficiente na modelagem de sistemas complexos.

# Objetivos da UML 2.0

- Além de manter os objetivos da UML 1.x, acrescenta-se:
  - Tornar a modelagem das entidades de software executáveis;
  - Prover mecanismos mais robustos para a modelagem de *workflow* e ações;
  - Criar padrões para a comunicação entre diferentes ferramentas (XMI).

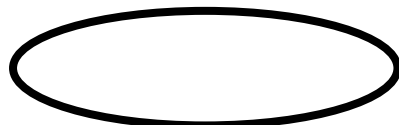
# Características da UML 2.0 – Elementos do Modelo

- Os conceitos utilizados nos diagramas.

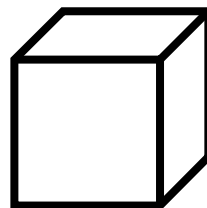
- Possuem:

- Semântica
- Representação gráfica
- Extensões

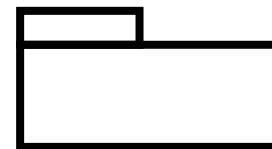
  
Composição



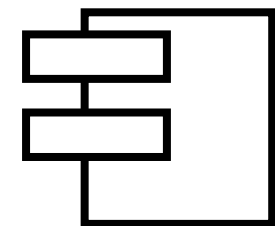
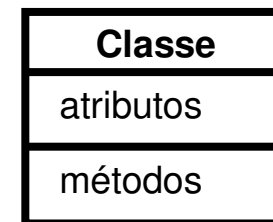
Caso de Uso



Nó



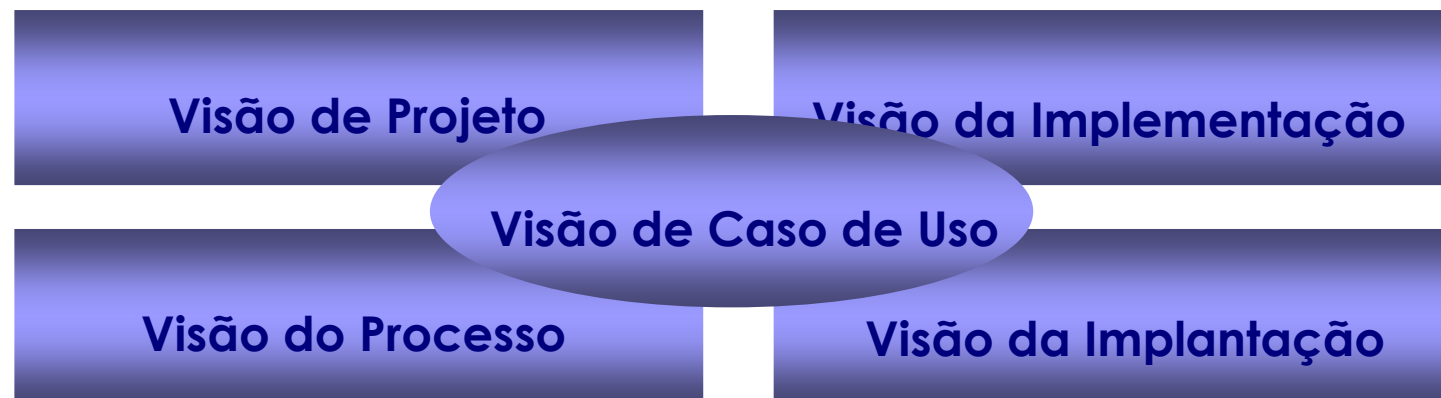
pacote



componente

# Características da UML 2.0 – Visões

- Mostram os diferentes aspectos do sistema.
- É uma abstração que contém vários diagramas.
- Os diagramas são associados a sua respectiva visão.
- Relacionam os modelos ao método ou processo utilizado para o desenvolvimento.



# Características da UML 2.0 - Diagramas

- Incluem os elementos gráficos que ilustram uma parte ou aspecto particular do sistema.
- Um modelo do sistema tem vários tipos de diagramas para descrevê-lo.
- Os Diagramas são:
  - Diagrama de Caso de Uso (Use Case)
  - Diagrama de Classe
  - Diagrama de Objetos
  - Diagrama de Estrutura de Compósito (Composite Structure)
  - Diagrama de Máquina de Estados

# Características da UML 2.0 - Diagramas

- Os Diagramas são:
  - Diagrama de Sequência
  - Diagrama de Comunicação
  - Diagrama de Atividade
  - Diagrama de Revisão da Interação (Overview Interaction)
  - Diagrama de Tempo (Timing)
  - Diagrama de Componente
  - Diagrama de Implantação (Deployment)

# Entendendo a Necessidade do Cliente

“A maior lição que a vida me ensinou  
é que às vezes, até os tolos têm  
razão.”

Wiston Churchill

# Objetivo

- Um sistema deve ter a capacidade de atender aos seus requisitos.
- Nosso problema é entender o problema do usuário dentro da sua cultura, linguagem e construir sistemas que venham de encontro às suas necessidades.
- Característica é um serviço que o sistema fornece a fim de atender as necessidades dos usuários

# Visão do Projeto

- É necessário entender o problema para saber o que deve ser solucionado.
- Passos:
  - Identificar os stakeholders;
  - Obter o problema a ser resolvido;
  - Definir os limites e restrições do sistema;
  - Formular o problema;
  - Definir as características do problema;
  - Avaliar os resultados;
  - Documentar no relatório “Visão do Projeto”.

# Documento de Visão

- Documenta a visão completa do sistema.
- Captura as expectativas entre os envolvidos.
- Escrito com base na perspectiva dos clientes.
- Objetivo nas características essenciais do sistemas e níveis aceitáveis de qualidade.
- Fornece uma base contratual para os requisitos visíveis dos envolvidos.
- Não é totalmente preenchida na Iniciação.

# Definindo o Escopo: Modelo de Caso de Uso

“Agradei-me, particularmente, da possibilidade de que Joshua estivesse tão preso ao seu modo clássico de pensar que me permitisse realizar o incrível feito de chegar antes do que ele à interpretação correta do seu próprio experimento.”

James Watson

# Modelo Caso de Uso

- É uma forma do engenheiro de requisitos especificar os limites e as funcionalidades do sistema.
- Permite:
  - Que clientes e usuários validem o sistema;
  - Que os desenvolvedores do sistema construam o que é esperado.
- Componentes:
  - Casos de usos
  - Atores

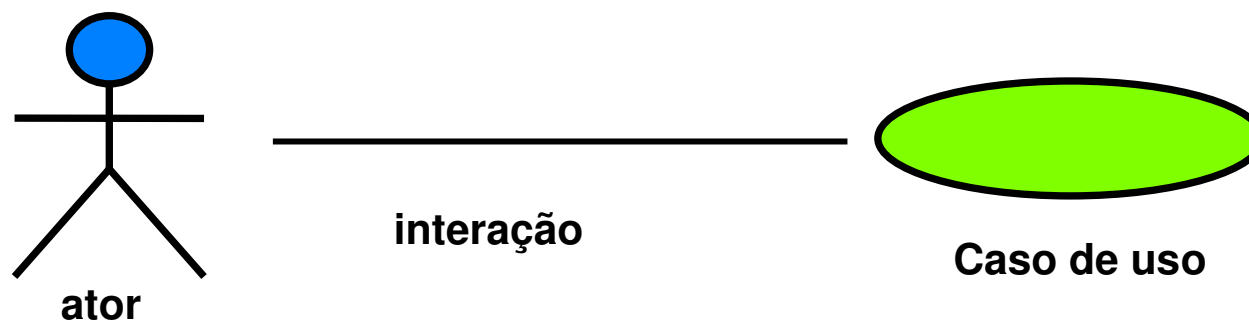
# Atores

- Atores são papéis de elementos externos ao sistema e que interagem DIRETAMENTE com o sistema.
- Um outro sistema que interage com o sistema a ser desenvolvido também é considerado um ator, desde que este sistema não faça parte do desenvolvimento.
- Exemplo de atores:
  - Cliente
  - Secretária
  - Sistema de vendas (desde que não seja o sistema que estamos desenvolvendo)
  - Glicosímetro (aparelho que mede a glicemia de uma pessoa, ele pode ser conectado ao computador por um cabo)

# Casos de Uso

- São funcionalidades que o sistema realiza e que fornece um benefício a um ator específico.
- As características do caso de uso são:
  - São sempre iniciadas por um ator.
  - Deve sempre retornar um resultado (valor) ao ator.
  - Cada caso de uso especifica uma funcionalidade completa envolvendo os atores interessados. Deve sempre terminar com o resultado que deve ser dado ao ator.

# Notação

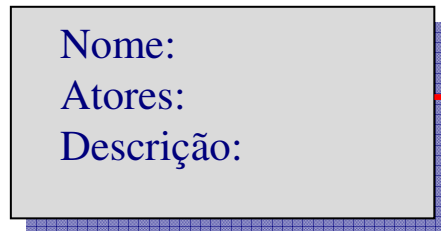


# Especificação do Casos de Uso

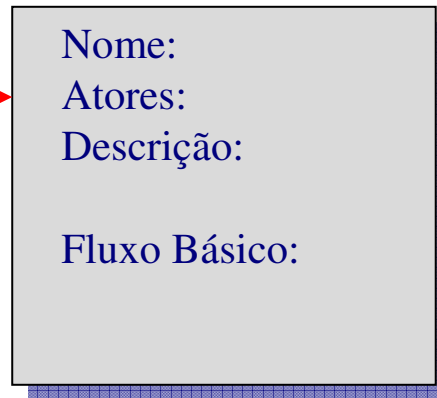
- Cada caso de uso no diagrama de casos de uso deve ser detalhado na especificação de casos de uso.
- Esta especificação é evolutiva, quanto mais requisitos são coletados, mais detalhes são adicionados na especificação.
- Os tipos de especificação são:
  - descrição inicial (representação conceitual do sistema);
  - descrição base (documenta o comportamento ideal);
  - descrição elaborada (documenta detalhadamente o comportamento).

# Tipos de Especificações de Caso de Uso

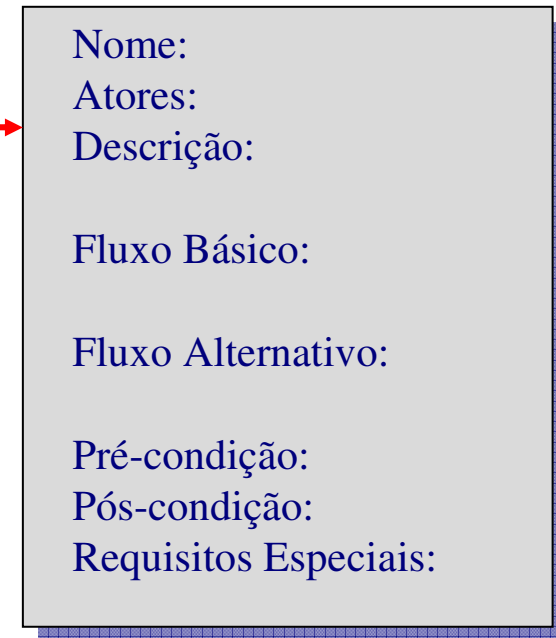
Descrição inicial



Descrição base



Descrição elaborada



# Refinar o Modelo de Casos de Uso

“Nada é mais fácil do que enganar a si mesmo. Pois, aquilo que cada homem deseja, isso mesmo ele acredita ser verdade.”

Demóstenes

# Refinar o Modelo de Casos de Uso

- Detalhar os casos de uso do sistema (Especificação do Caso de Uso)
  - Detalhar o fluxo de evento dos casos de uso
    - *Fluxo Básico: Quando a atividade é realizada com sucesso. Deve existir somente uma.*
    - *Fluxo Alternativo: Quando a atividade não é realizada com sucesso o que deve acontecer. Pode existir quantas forem necessárias (cobrir todas as situações).*
  - Atividades (para cada caso de uso)
    - Fluxo Básico (“Caminho Feliz”)
      - Identificar ações
      - Numerar ações em sequência ao qual acontecem
    - Fluxo Alternativo (“Exceções”)
      - Identificar ações
      - Numerar ações em sequência ao qual acontecem
  - Estruturar o fluxo de eventos dos casos de uso – diagrama de atividade

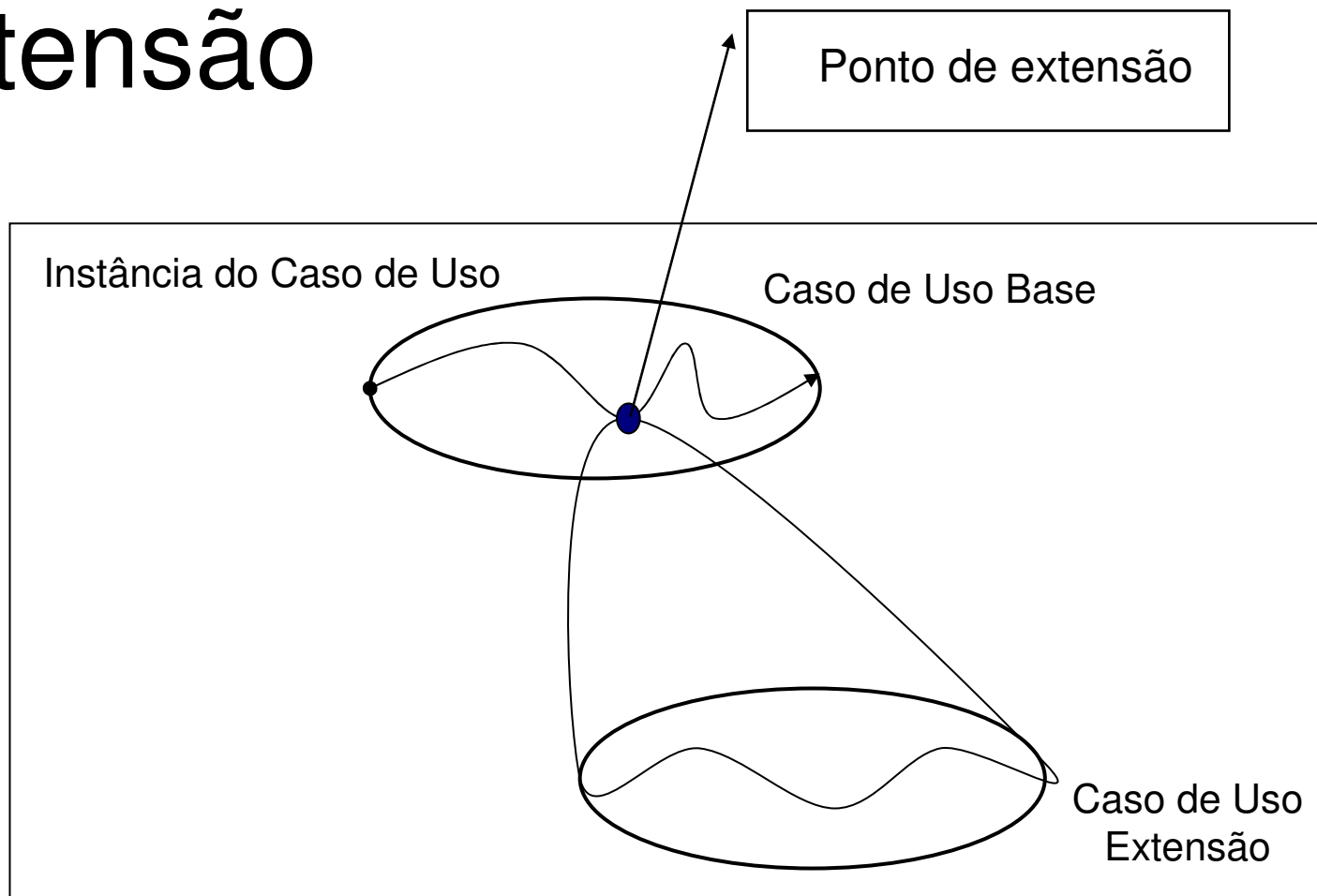
# Refinar o Modelo de Casos de Uso

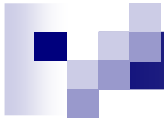
- Detalhar os casos de uso do sistema
  - Descrever os requisitos especiais:
    - Requisitos legais e de regulamentação
    - Padrões de aplicativos
    - Requisitos de usabilidade, confiabilidade, desempenho e suportabilidade
    - Sistemas operacionais, ambientes, compatibilidade e restrições de projeto.
  - Descrever a pré, pós condição e pontos de extensão se pertinente.

# Refinar o Modelo de Casos de Uso

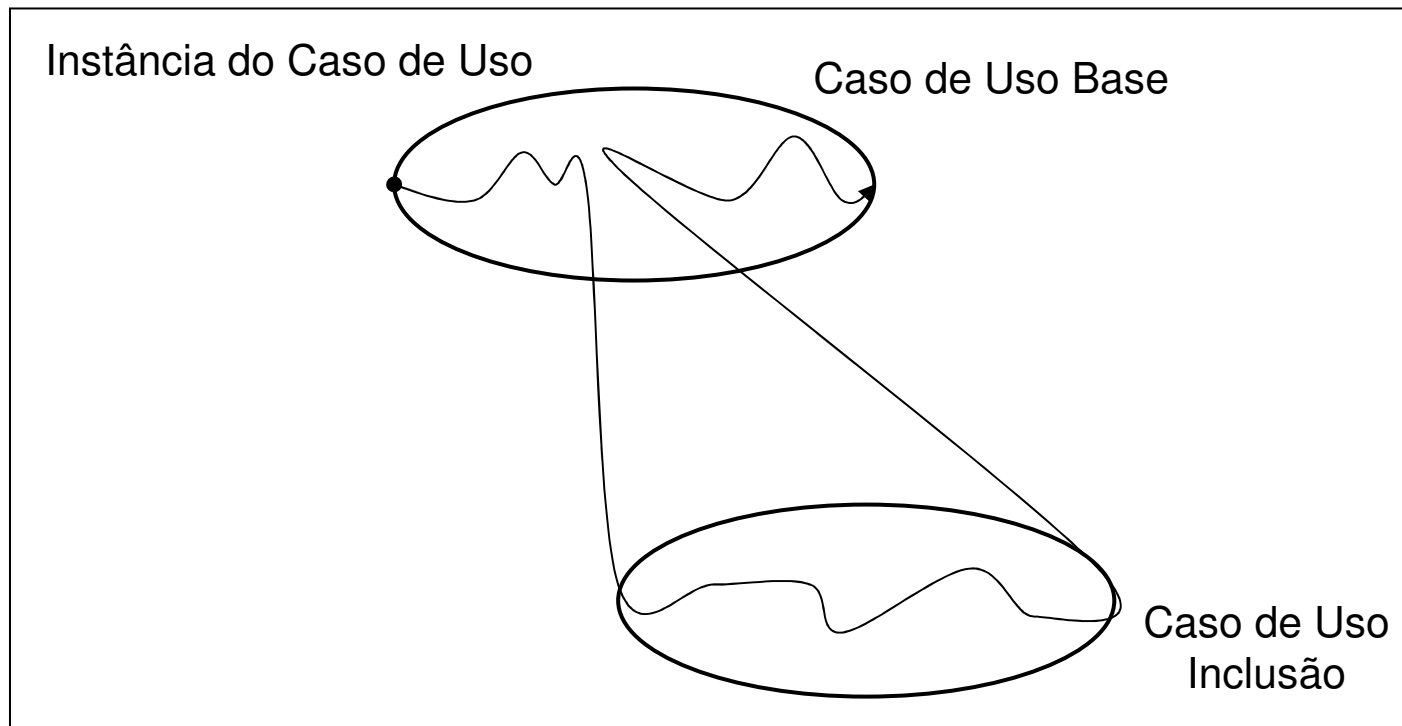
- Estruturar modelo de casos de uso
  - Estabelecer relacionamento de “Inclusão” entre os casos de uso
  - Estabelecer relacionamento de “Extensão” entre os casos de uso
  - Estabelecer relacionamento de “Generalização” entre os casos de uso
  - Estabelecer relacionamento de “Generalização” entre os atores

# Extensão

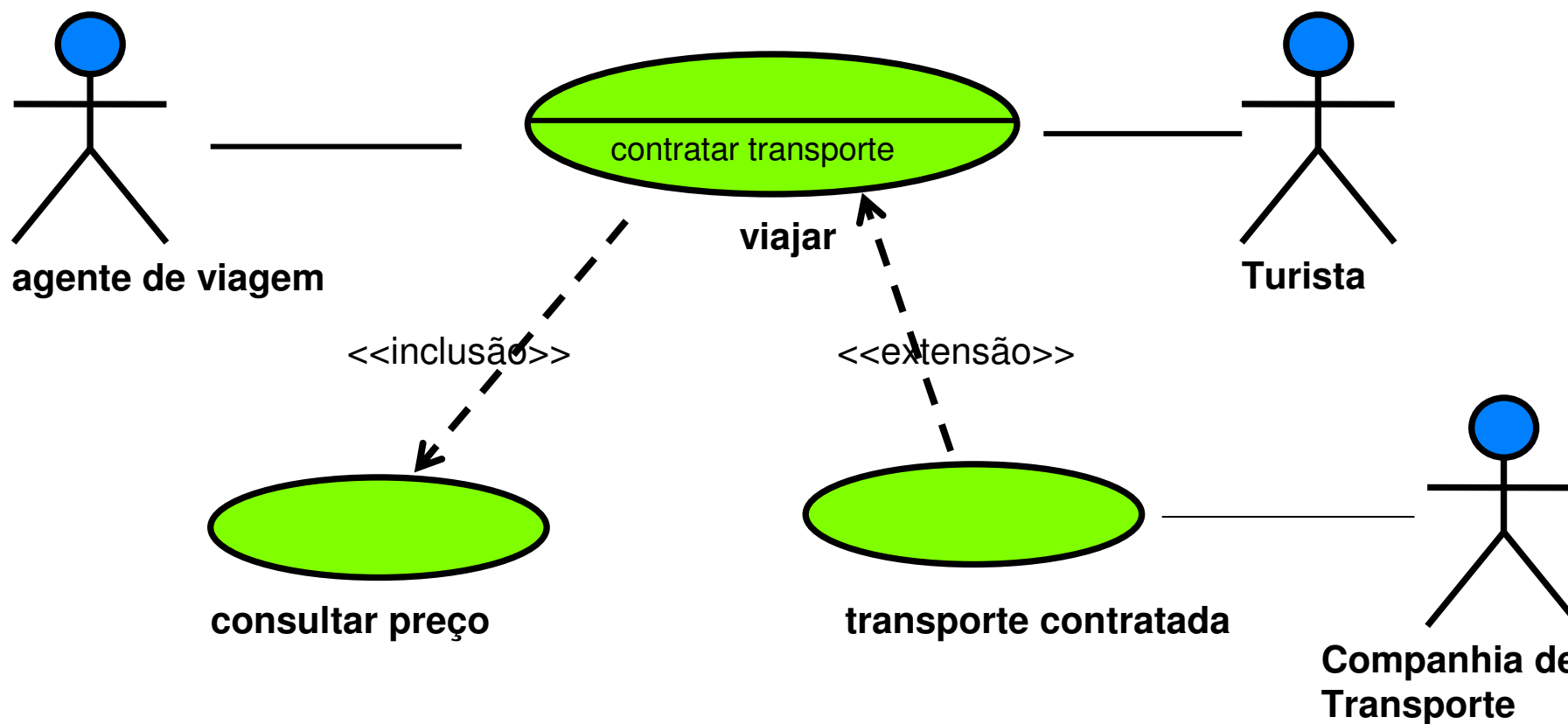




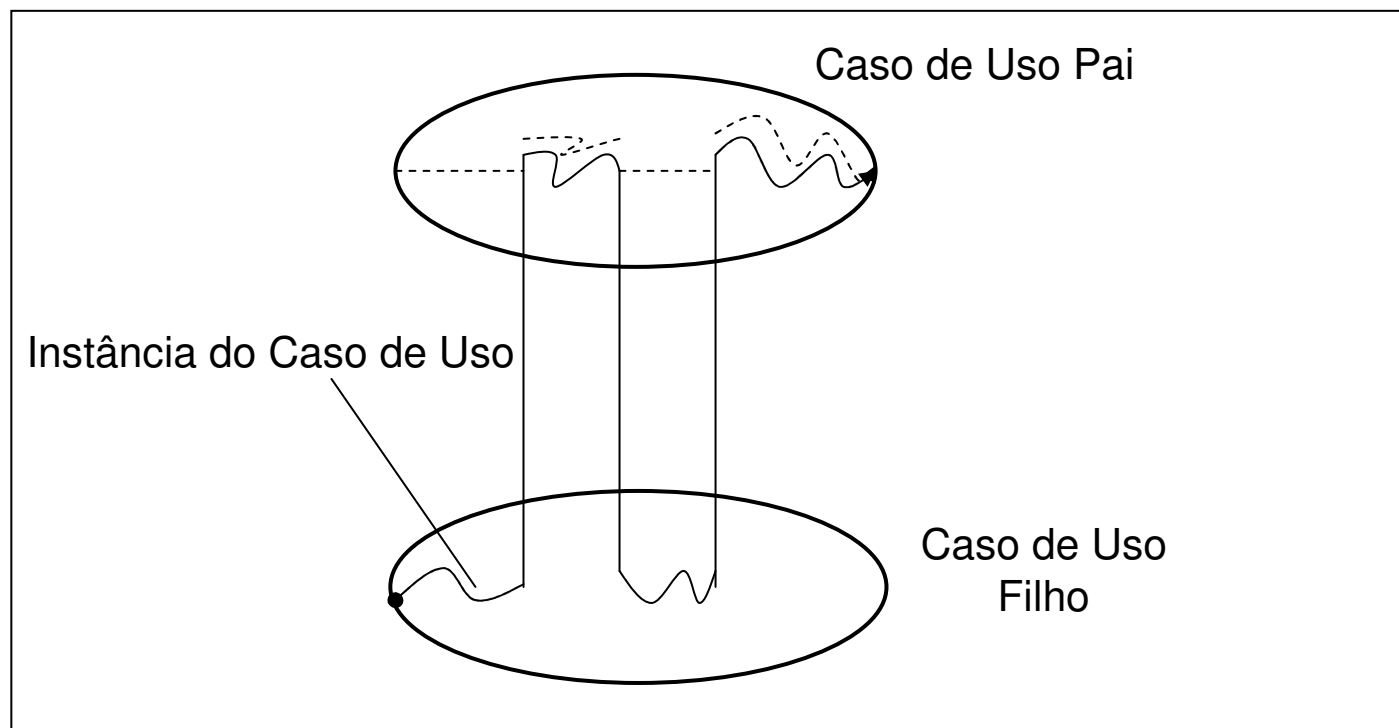
# Inclusão



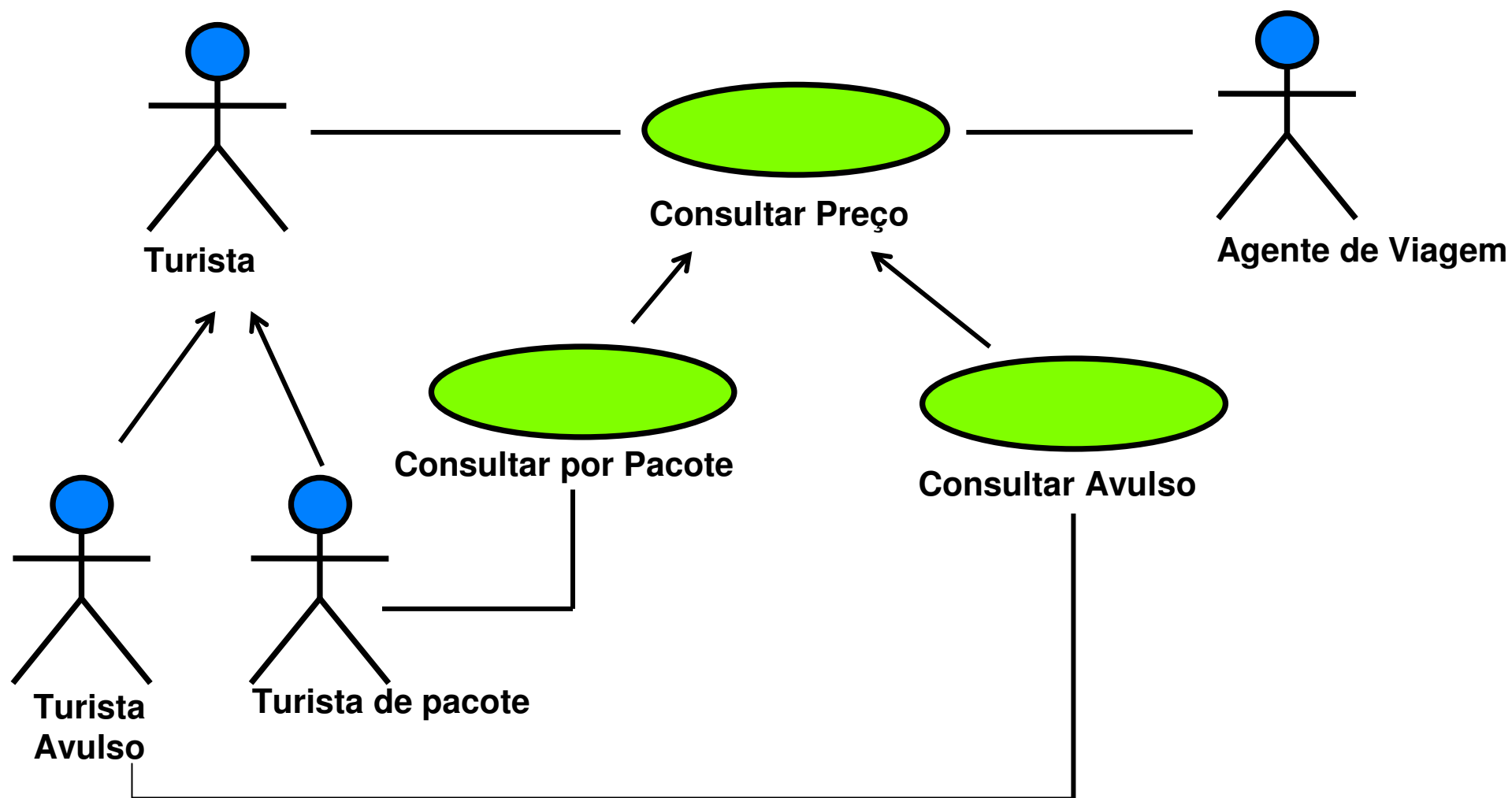
# Notação



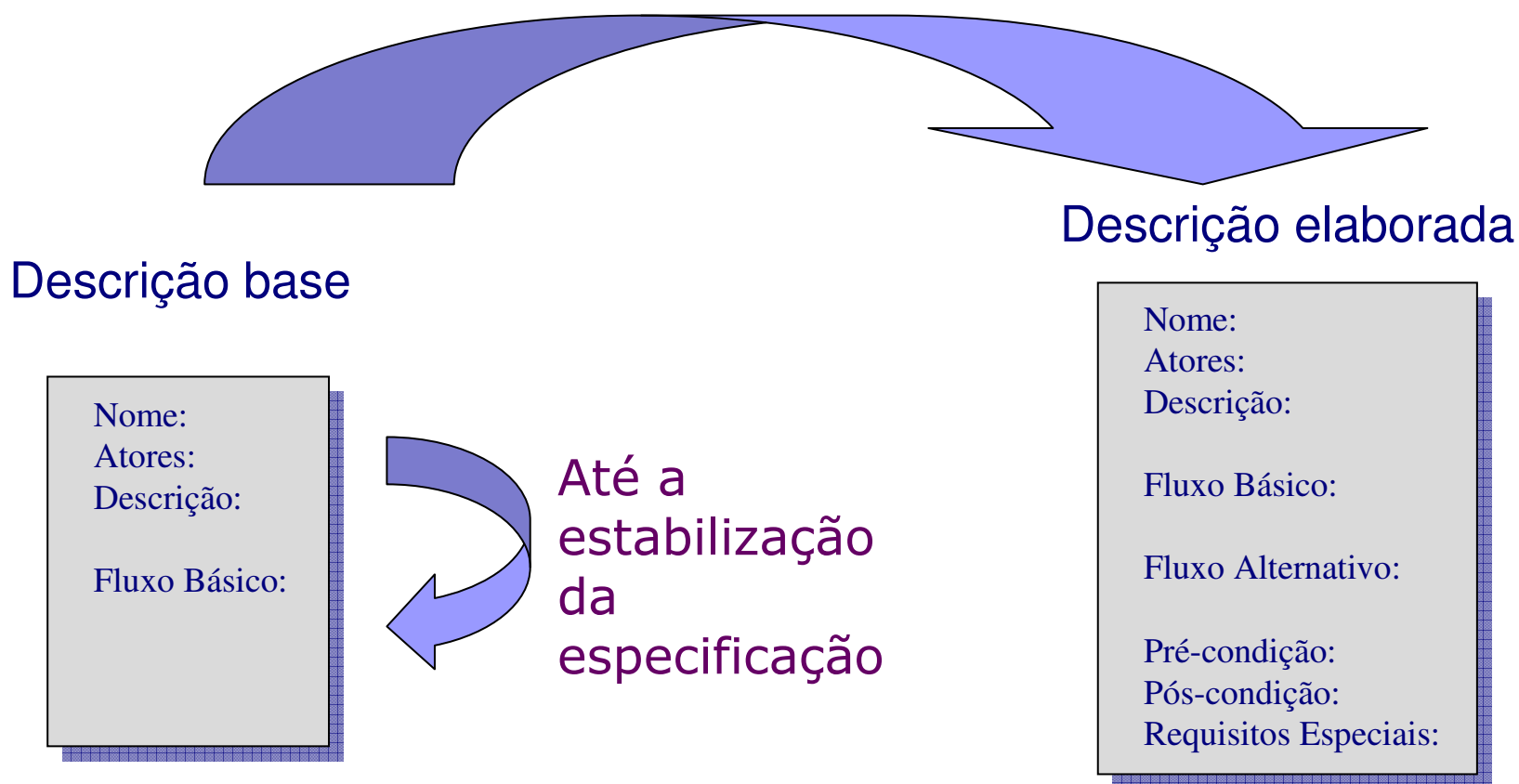
# Generalização



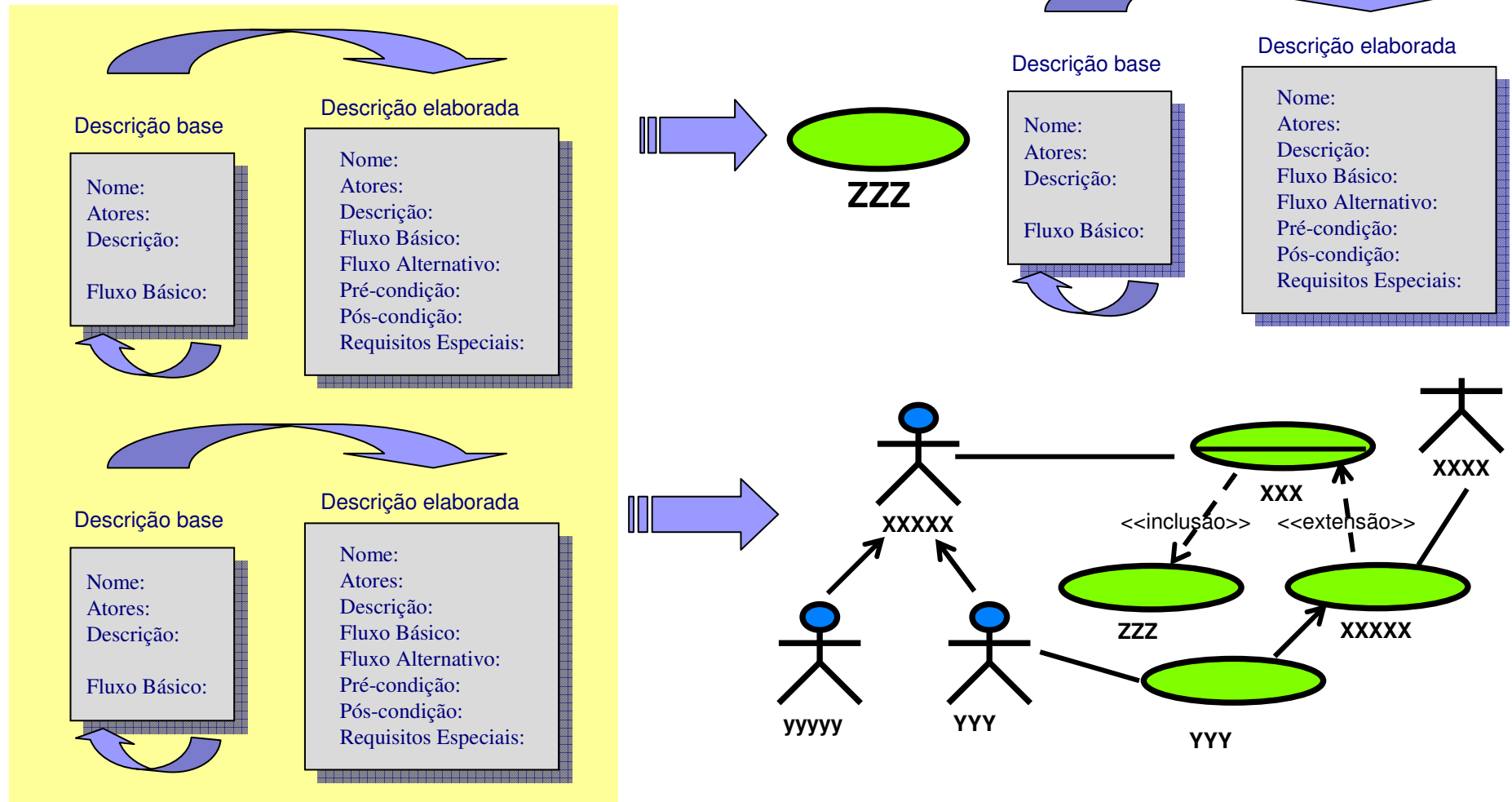
# Notação



# Detalhar os casos de uso - Dinâmica



# Estruturar o modelo de caso de uso - Dinâmica

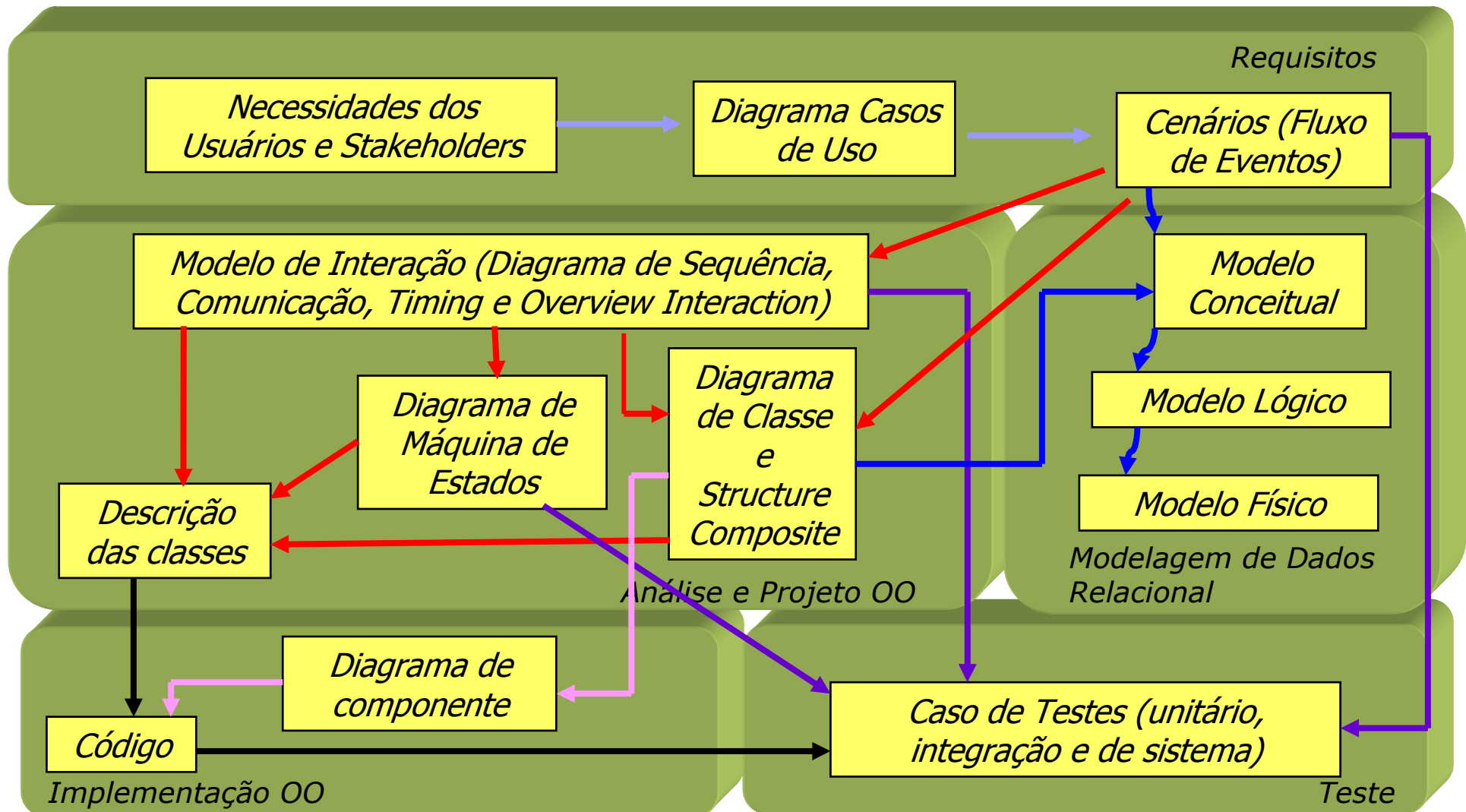


# E depois?

“A única razão da existência do tempo é para que todas as coisas não aconteçam de uma vez.”

Albert Einstein

# Desenvolvimento de Sistemas e a UML



# Pensamentos Finais

"O que você fizer será insignificante, mas é muito importante que você o faça."

Mahatma Gandhi

"É muito mais fácil fazer críticas do que fazer o que é certo."

Benjamin Disraeli

# Bibliografia

- The Unified Modeling Language User Guide - Grady Booch, James Rumbaugh, Ivar Jacobson - Addison Wesley – 1999
- UML and the Unified Process – Jim Arlow, Ila Neustadt – Addison Wesley – 2003.
- The Unified Software Development Process - Ivar Jacobson, Grady Booch, James Rumbaugh - Addison Wesley - 1999.
- UML 2 Toolkit - Hans-Erik Eriksson e Magnus Penker - Wiley Computer Publishing – 2004
- Writing Effective Use Cases - Alistair Cockburn - Addison Wesley - 2002.
- Rational Software Corporation; Rational Unified Process Version 2002.05.20; Copyright 1987 – 2002; 2002.

# UML 2.0 - Modelo Casos de Uso

*Márcia Ito*

[ito@mind-tech.com.br](mailto:ito@mind-tech.com.br)

*Julho/2004*