



Análise e Projeto de Sistemas I

Introdução UML

- Linguagem Unificada de Modelagem (UML) do OMG.
- Versão 1.5.





Objetivos.

- Modelar sistemas (não somente software) utilizando conceitos de Orientação a Objetos.
- Estabelecer uma clara ligação entre os modelos conceitual e de implementação.
- Explicitar os pontos notáveis em sistemas complexos ou de missão crítica.
- Definir uma linguagem de modelagem passível de uso por pessoas e máquinas.



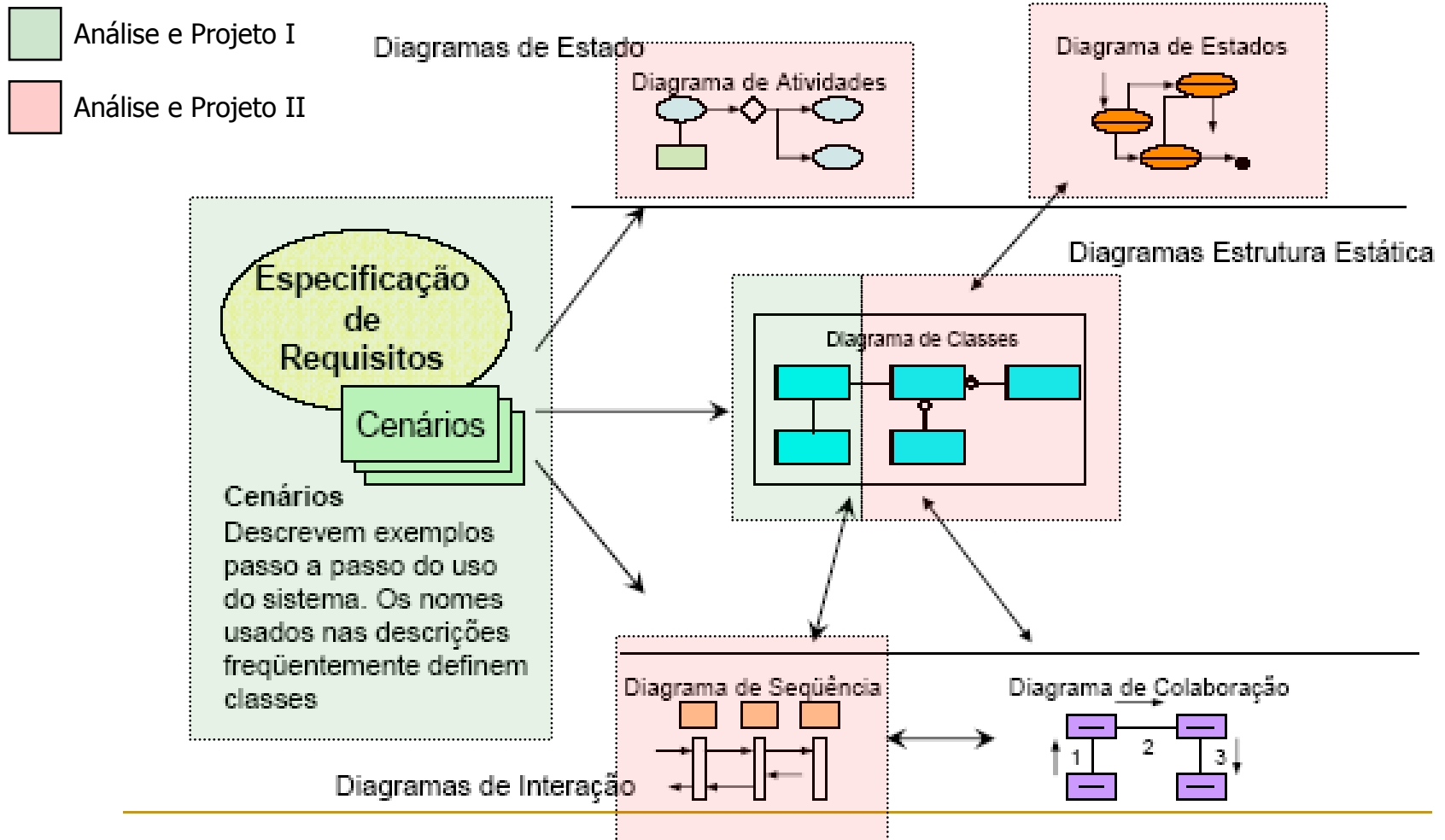
Características.

- Independente de método, de ferramenta, de linguagem de programação e de metodologia de desenvolvimento.
- Suporta conceitos de desenvolvimento de alto nível, bem como conceitos como colaboração, *framework*, padrões de *design* e componentes.
- Bastante completa e extensa – aproximadamente 40% da linguagem cobre 98% das necessidades de um projeto comum.

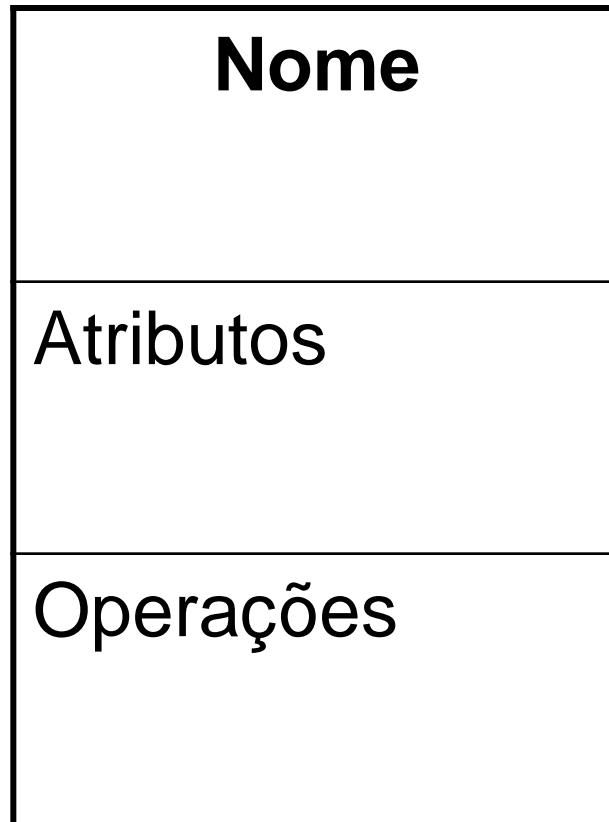
Conceitos básicos.

- Visões de modelagem UML.
 - Mostram diferentes aspectos do sistema modelado.
- Diagramas de modelagem UML.
 - Gráficos (13 tipos) que descrevem o conteúdo de uma visão.
 - Uma visão pode ser composta por vários diagramas.
- Elementos de modelagem UML.
 - Representam os conceitos da Orientação a Objetos, como classes, objetos, mensagens e as relações entre esses conceitos
- Mecanismos de modelagem UML.
 - Fornecem comentários, informações ou explicações sobre os elementos.

Visão esquemática.



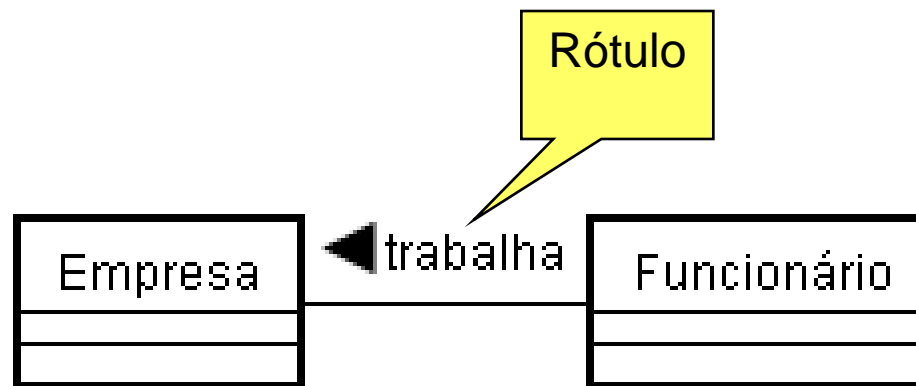
Notação para classe.



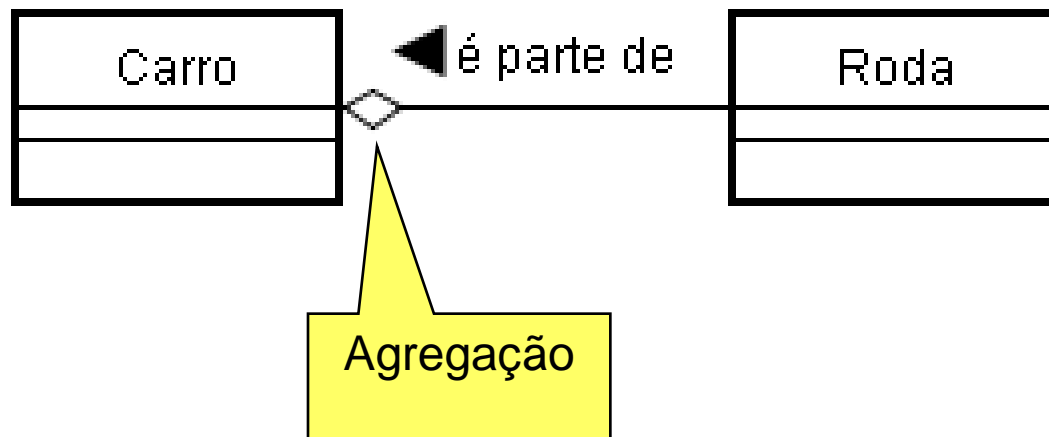
Notação para objeto.

<u>Identificação:Nome da classe</u>
Atributos = valores

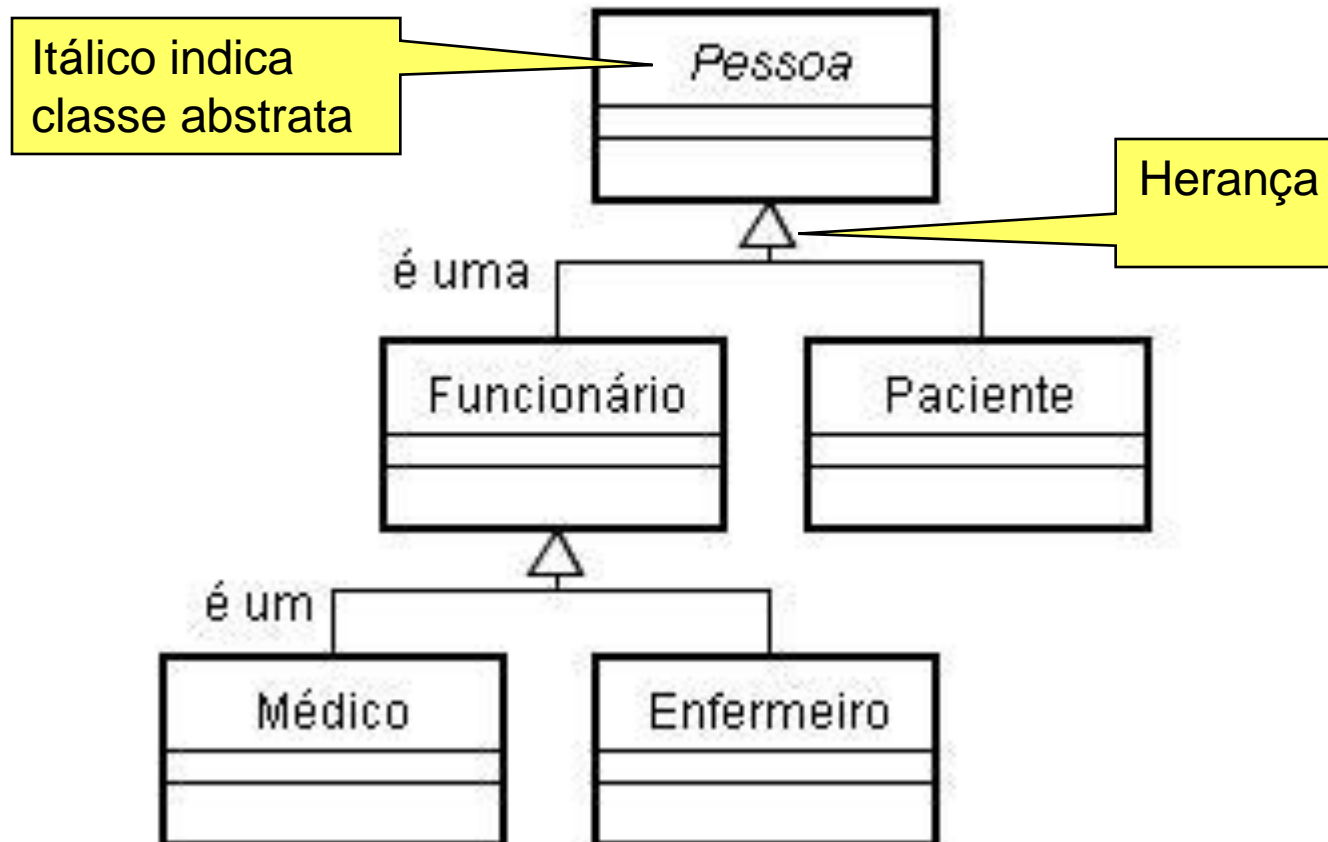
Associação.



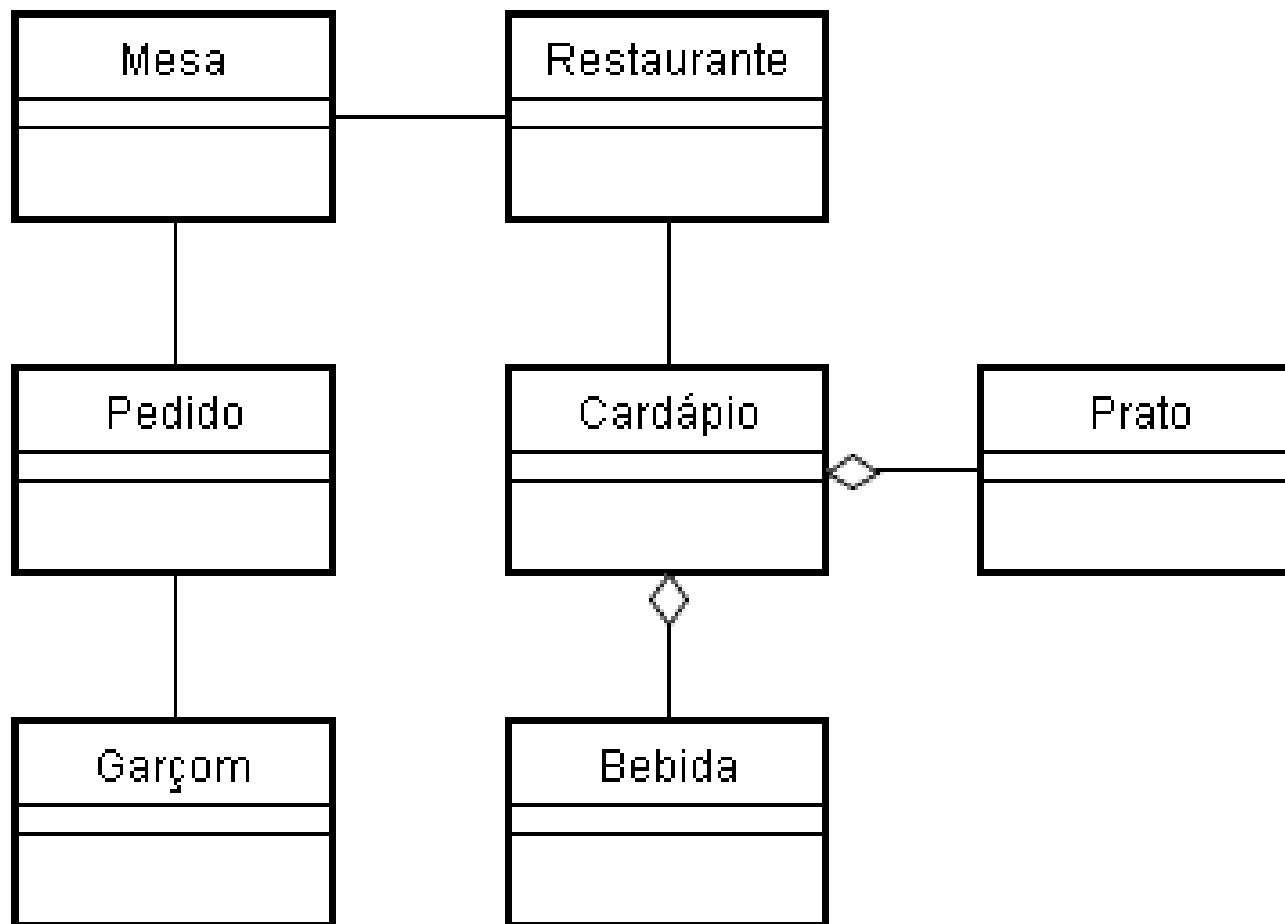
Agregação.



Herança.



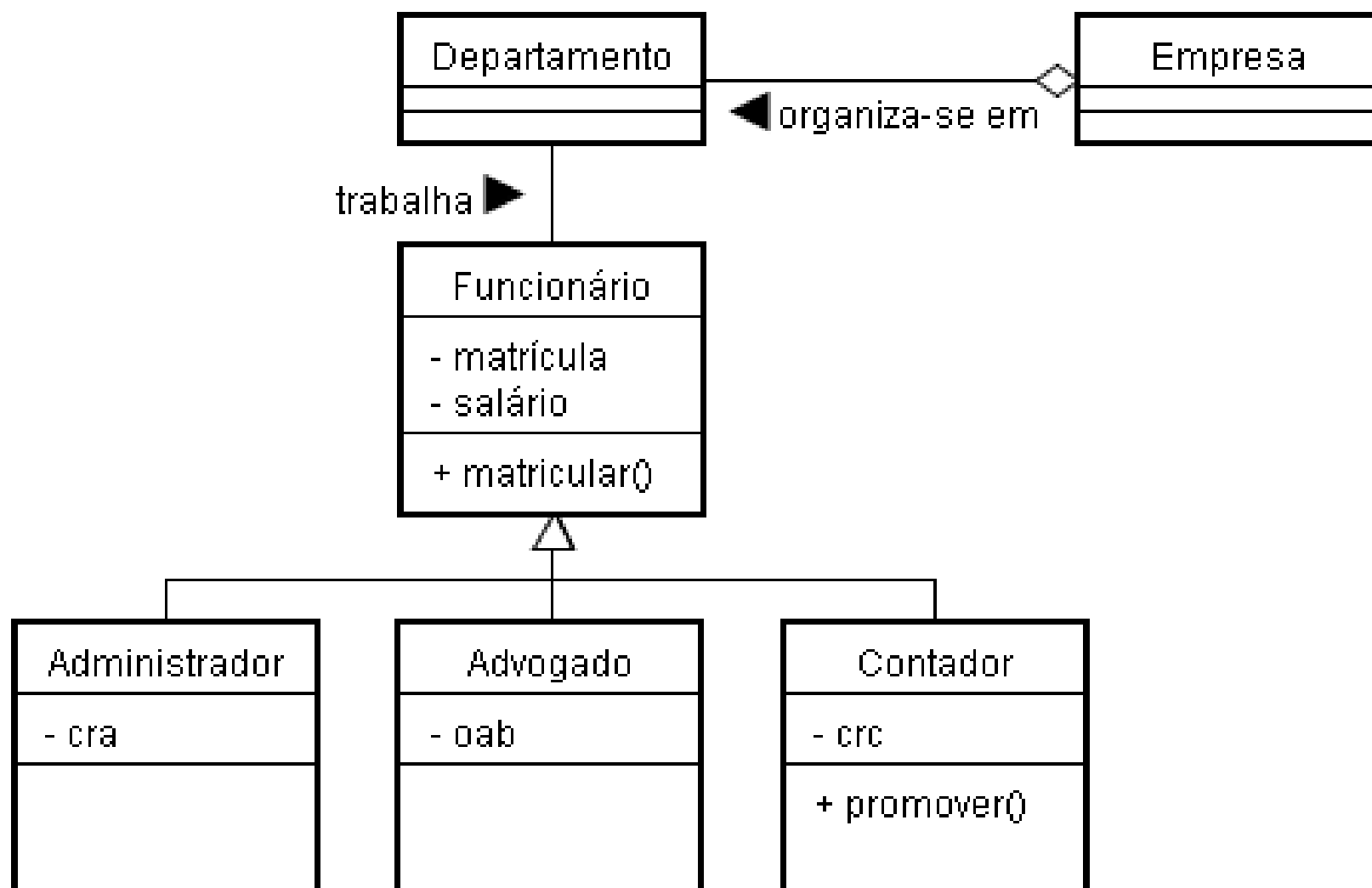
Exercício 1 – diagrama.



Exercício 1 – questões.

- Interprete o diagrama adicionando rótulos.
- Inclua a classe Ingrediente no modelo, associando-a adequadamente.
- Adicione dois atributos a cada classe.
- Associe a classe Garçon à classe Mesa e reinterprete o diagrama.

Exercício 2 – diagrama.



Exercício 2 – questões.

- Interpretando o diagrama acima, avalie as afirmativas abaixo como verdadeiras ou falsas:
 - ☐ Departamentos fazem parte da empresa.
 - ☐ A empresa só possui funcionários administradores, contadores e advogados.
 - ☐ Os funcionários são lotados nos departamentos da empresa.
 - ☐ Advogado é um tipo de funcionário que se diferencia dos demais pelo atributo OAB.
 - ☐ Administradores são funcionários sem o atributo matrícula, mas com o atributo exclusivo CRA.
 - ☐ Matricular é uma operação polimórfica.
 - ☐ Abaixo está representado um objeto da classe Contador.

<u>maria:Contador</u>
matrícula = 123 salário = 100,00 oab = 5714



Requisitos de Software com UML



Conceitos importantes.

- Requisitos:

- Uma condição ou capacidade à qual um software deve apresentar conformidade.

- Requisitos funcionais:

- Definem ações que um software deve ser capaz de oferecer, sem levar em consideração restrições físicas.

- Requisitos não-funcionais:

- Demais requisitos, principalmente, tecnológicos e de qualidade.

Por que requisitos de software são importantes?

- O desafio está em entender a necessidade do usuário (respeitando sua cultura e sua linguagem) para construir softwares que venham ao encontro de suas expectativas.
- Precisamos descobrir as funcionalidades que o software deverá apresentar para atender nossos usuários.
- Essas funcionalidades (ou requisitos funcionais) formam um “contrato” com os usuário, contendo tudo o que o software propiciará fazer.
- O conjunto de todos os requisitos funcionais explícitos aos quais o software deverá apresentar aderência é definido como o escopo do *problema*.
- Todo requisito funcional que não tenha sido explicitamente solicitado pelos usuários não pertence ao escopo e, portanto, não deve ser atendido.

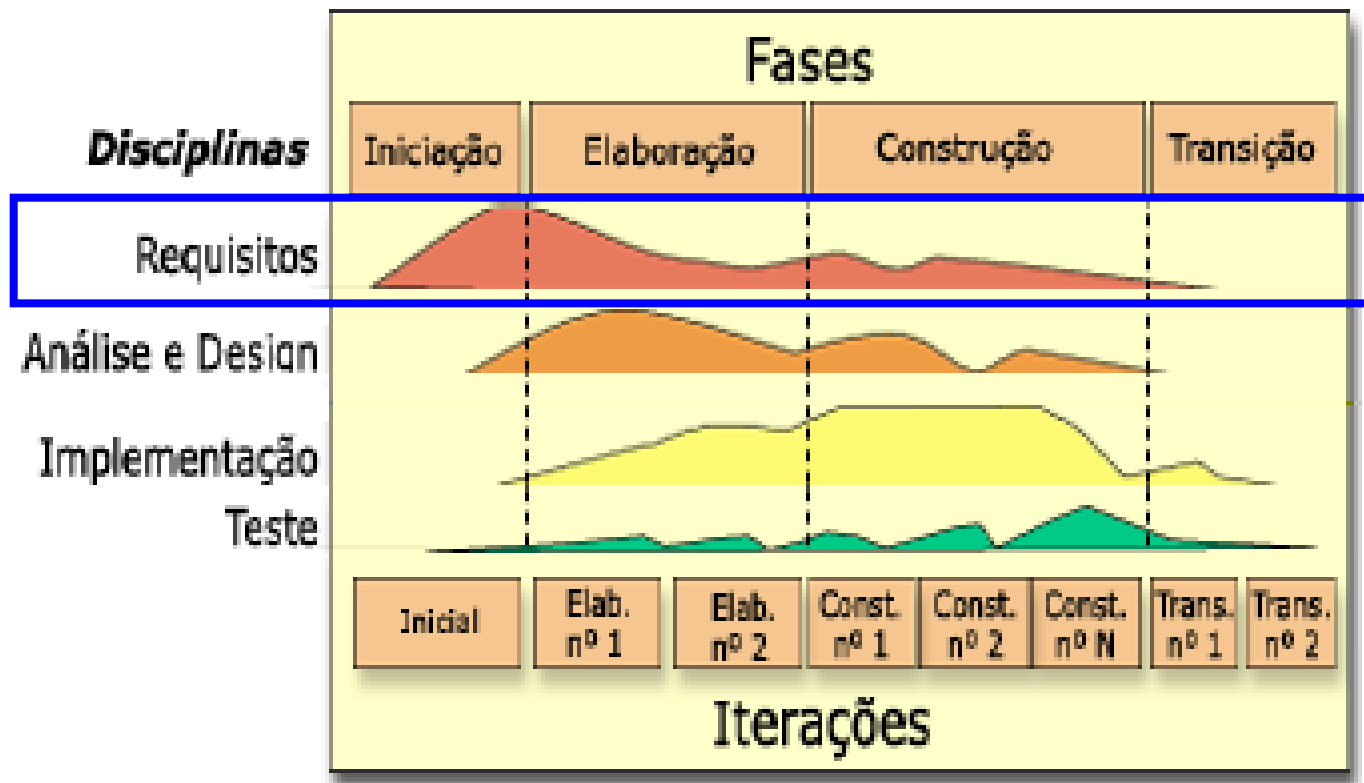
Mais perguntas...

- Qual o caminho para descobrir os requisitos?
- Quais as dificuldades existentes?
- Como encontrá-los e registrá-los de maneira a não nos perdermos diante de tudo o que há para desenvolver?



Disciplina Requisitos.

Requisitos no RUP.





Objetivos.

- Estabelecer e manter acordo com os usuários e outros envolvidos sobre o que o software deverá fazer.
- Disponibilizar para os desenvolvedores um melhor entendimento dos requisitos do software.
- Definir as fronteiras (delimitações) do software.
- Prover uma base para o planejamento do conteúdo técnico das iterações.
- Prover uma base para estimar o custo e o tempo para desenvolver o software.
- Definir uma interface de usuário para o software, focando nas necessidades e metas dos usuários.

Estratagemas.

	1º Passo	2º Passo
Meta	Estabelecer o problema	Definir a solução
Técnica	Entrevistas	Entrevistas e modelagem de casos de uso
Artefato(s)	Visão e Glossário	Modelo de caso de uso



Dificuldades ao lidar com requisitos.

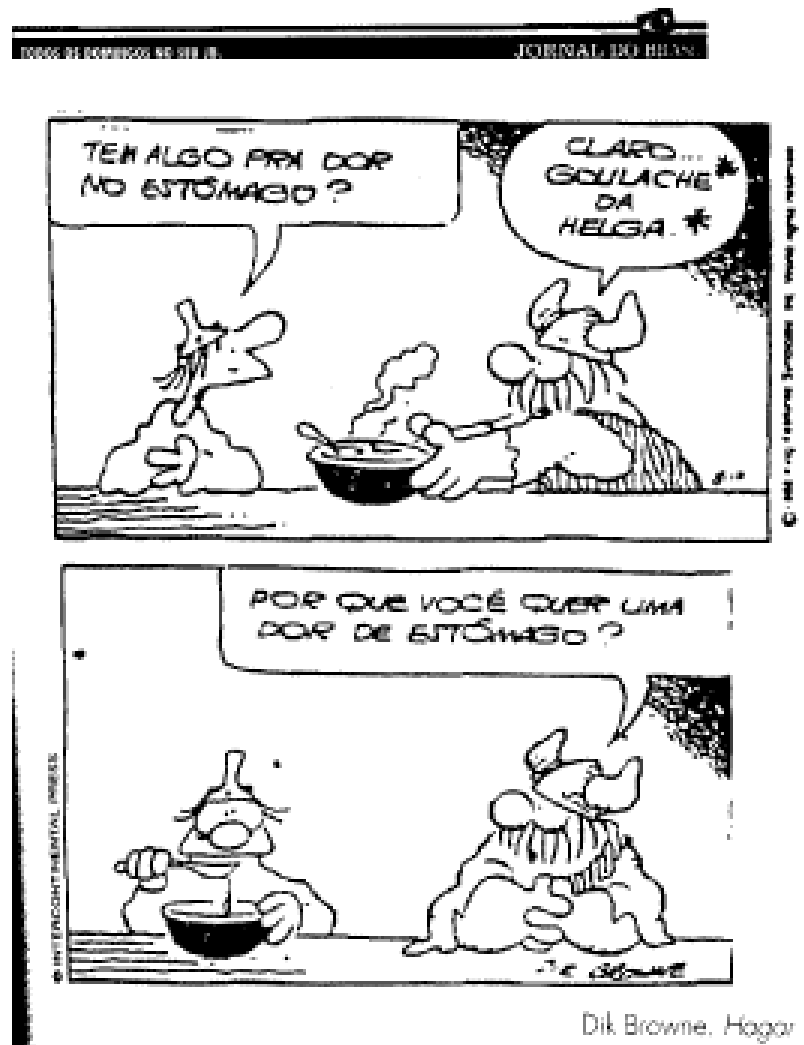
- Comunicação humana:
 - Nossa maior aliada é, ao mesmo tempo, nossa maior fraqueza (paradoxo!).
- Representação das idéias:
 - Temos que expressá-las da forma mais precisa possível.
 - Precisamos de uma linguagem adequada.

Dificuldades da comunicação (1).

- *“Sei que você acredita que entendeu o que acha que eu disse, mas não estou certo de que percebe que aquilo que ouviu não é o que eu pretendia dizer...”* (Cliente anônimo).

Citado por Roger Pressman.

Dificuldades da comunicação (2).



Dificuldades da comunicação (3).



How the customer explained it



How the Project Leader understood it



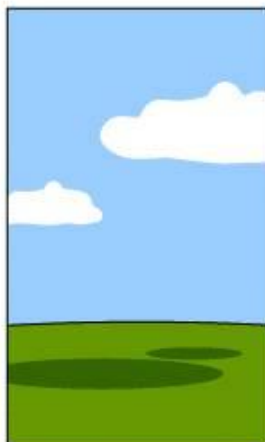
How the Analyst designed it



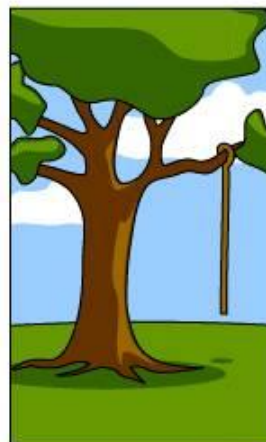
How the Programmer wrote it



How the Business Consultant described it



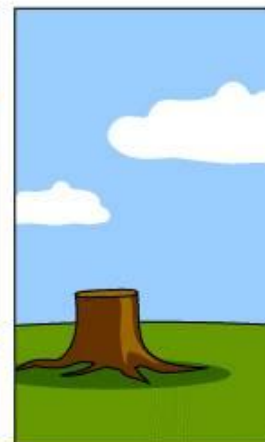
How the project was documented



What operations installed



How the customer was billed



How it was supported



What the customer really needed



Técnicas de entrevista*.

(*) No contexto da Engenharia de Requisitos de Software.

Objetivos.

- Coletar informações sobre os requisitos de um novo sistema (ou sobre o comportamento de um sistema atual) junto às pessoas que têm essas informações armazenadas em suas mentes.
- Verificar nossa própria compreensão, enquanto analistas de sistemas, dos requisitos de um novo sistema (ou do comportamento de um já existente).

Procedimentos.

- Selecionar entrevistados:
 - Para evitar desperdício de tempo e desgaste do projeto.
 - Basear-se no organograma da empresa.
 - Definir níveis: operativo, supervisão operativa e gerencial.
- Planejar entrevistas:
 - Selecionar técnicas e ferramentas – gravador, questionários, etc.
 - Definir roteiro de perguntas.
 - Agendar entrevistas, mencionando motivo, assunto, data e hora, duração e local, e anexar lista de perguntas.
- Entrevistar:
 - Seguir planejamento, mas permitir improvisações.
- Redigir relatório resultante da entrevista.
- Validar relatório com entrevistados (se necessário, em uma nova entrevista).

Exemplos de técnicas.

- Reunião pessoal e direta.
 - Reuniões envolvendo os analistas e os usuários.
 - Realizadas em diversas ocasiões, de acordo com a especificidade de atuação de cada usuário.
- IBM JAD (*Joint Application Development*).
 - *Workshop*, organizado em sessões, que pode-se prolongar por vários dias e das quais são extraídos todos os requisitos dos usuários.
- FAST (*Facilitated Application Specification Techniques*).
 - Série de encontros do tipo *brainstorming* entre pessoas envolvidas (*stakeholders*) no projeto de desenvolvimento do software.



Caso de uso.

- Unidade coerente de funcionalidade disponibilizada por um sistema, representada por uma seqüência de mensagens trocada entre este e um agente externo ao sistema e suas ações.
- É uma narrativa que descreve a seqüência de eventos de um ator (um agente externo) utilizando o sistema para executar um processo.
- Não se propõe a ser uma especificação de requisitos ou especificação funcional formal, mas serve para ilustrar e elucidar os requisitos do sistema.
- Representa uma funcionalidade do sistema, como vista pelo usuário.



Ator.

- Externo ao sistema; presente no ambiente.
- Papel de alguém ou de alguma coisa.
- Abstração de alguma pessoa ou coisa que utiliza o sistema, inclusive outro sistema.
- Tudo aquilo que interage com o sistema e que interessa ser modelado.

Diagrama UML de caso de uso.

- Elementos e notação:





Como identificar atores.

■ Deve-se observar:

- ☐ Quem utiliza diretamente o sistema.
- ☐ Quem é responsável por manter o sistema.
- ☐ Hardware externo usado pelo sistema.
- ☐ Outros sistemas que precisam interagir com o sistema.



Como identificar casos de uso.

- A partir dos atores:
 1. Identificar os atores relacionados ao sistema ou à organização.
 2. Para cada ator, identificar os processos que eles iniciam ou em que participam.
- A partir dos eventos:
 1. Identificar os eventos externos aos quais o sistema deve responder.
 2. Relacionar os eventos a atores e a casos de uso.



Heurísticas para casos de uso.

- Caso de uso é um fluxo específico de eventos para o sistema com possibilidades de ocorrência similares.
- Estes eventos ocorrem em uma mesma ocasião.
- Diversas alternativas derivadas desses eventos são melhor agrupadas em um único caso de uso.
- Um caso de uso deve produzir um resultado mensurável.
- Deve-se modelar a interação sob o enfoque do ator, ou seja, sem apresentar detalhes internos do funcionamento do sistema de informação.

Fluxos alternativos e condições.

- Fluxos alternativos.

- ☐ Abordam o comportamento de caráter opcional ou excepcional em relação ao comportamento normal (fluxo típico de eventos) e também as variações do comportamento normal.
- ☐ São "desvios" do fluxo de eventos básico, alguns dos quais voltarão ao fluxo de eventos básico e alguns finalizarão a execução do caso de uso.

- Pré-condições.

- ☐ Explicam o estado em que o sistema deve estar para que seja possível iniciar o caso de uso.

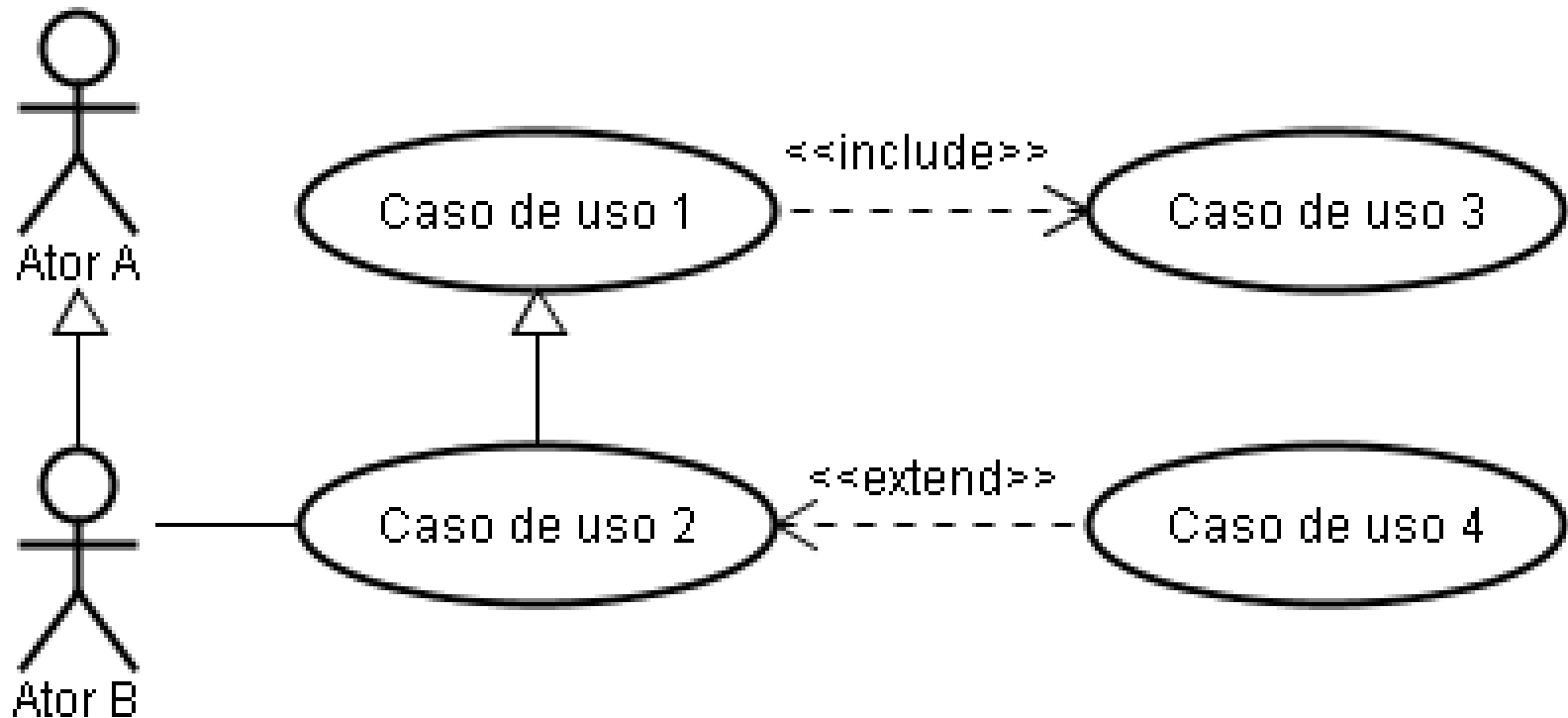
- Pós-condições.

- ☐ Mostram os estados em que o sistema pode estar ao final do caso de uso. O sistema deve assumir compulsoriamente um desses estados após a execução do caso de uso.

Associações permitidas – resumo.

	Comunicação	Generalização	Inclusão	Extensão
Ator-caso de uso	Permitida.			
Ator-ator		Permitida.		
Caso de uso-caso de uso		Permitida.	Permitida.	Permitida.

Associações permitidas – notação.



Associações permitidas – semânticas.

- Comunicação ator-caso de uso.
 - Atores que participam, utilizam o caso de uso.
- Generalização ator-ator.
 - Atores que apresentam características comuns.
- Generalização caso de uso-caso de uso.
 - Ao ocorrer uma instância do caso de uso filho, ela deve seguir tanto a descrição do filho quanto a do pai, para, só então, ser considerada realizada.
- Inclusão caso de uso-caso de uso.
 - Uma instância do caso de uso base, para ser concluída, também precisa seguir a descrição do caso de uso incluído.
- Extensão caso de uso-caso de uso.
 - Uma instância do caso de uso base, para ser concluída, alternativamente pode seguir a descrição do caso de uso extensor.

Artefatos para prototipar interfaces com o usuário.

- Guia de interface do usuário.
 - Normalmente fornecido pelo cliente, contém orientações sobre apresentação de elementos visuais e ergonômicos para possibilitar uma interação amigável do usuário com o sistema.
- Encenação de caso de uso.
 - Descrição lógica e conceitual de como um caso de uso é fornecido pela interface do usuário, incluindo a interação necessária entre os atores e o sistema.
- Classe de fronteira.
 - Representa uma interface entre o sistema e alguma entidade fora dele: uma pessoa ou outro sistema.
 - Usada para mediar o intercâmbio de informações com o mundo externo e isolar o sistema de mudanças ao seu redor.
- Protótipo da interface do usuário.
 - Proposta da interface do software, podendo ser esboços em papel, figuras *bitmaps* feitas com uma ferramenta de desenho ou protótipo de executável interativo feito com alguma ferramenta RAD.