



INSTITUTO DE CIÊNCIAS EXATAS

DEPARTAMENTO DE COMPUTAÇÃO

# UML 2.X

Eduardo Kinder Almentero  
[ekalmentero@gmail.com](mailto:ekalmentero@gmail.com)

# Introdução

- OMG
  - Fundado em 1989, o *Object Management Group, Inc.* (OMG) é uma associação aberta, sem fins lucrativos que produz e mantém especificações da indústria de computadores para dispositivos interoperáveis, portáteis e aplicativos corporativos reutilizáveis em ambientes distribuídos e heterogêneos.
- Documentos importantes:
  - Especificação OMG da UML 2.5.1 (Dezembro de 2017)
    - <https://www.omg.org/spec/UML/> (acesso em fevereiro 2022)
  - Definição de Diagrama da OMG 1.1 (*Diagram Definition* - DD)
    - <https://www.omg.org/spec/DD/> (acesso em acesso em fevereiro 2022)
    - Estabelece uma base para modelagem e intercâmbio entre diagramas com notações gráficas, principalmente os de estilo nó-arco, como os da UML, SysML e BPMN.

# Como surgiu a UML?

Grady BOOCH  
(1992, revisado em 1994)

BOOCH

- ✓ Diagrama de Estados
- ✓ Diagrama de Objetos
- ✓ Diagrama de Processo
- ✓ Diagrama de Módulos

Object-Modeling  
Technique (1991)

OMT

- ✓ Diagrama de Estados
- ✓ Diagrama de Classes

Object-Oriented  
Software  
Engineering (1992)

OOSE

- ✓ Use Case
- ✓ Subsistemas (Package)
- ✓ Diagrama de Interações
- ✓ MiniEspecificação

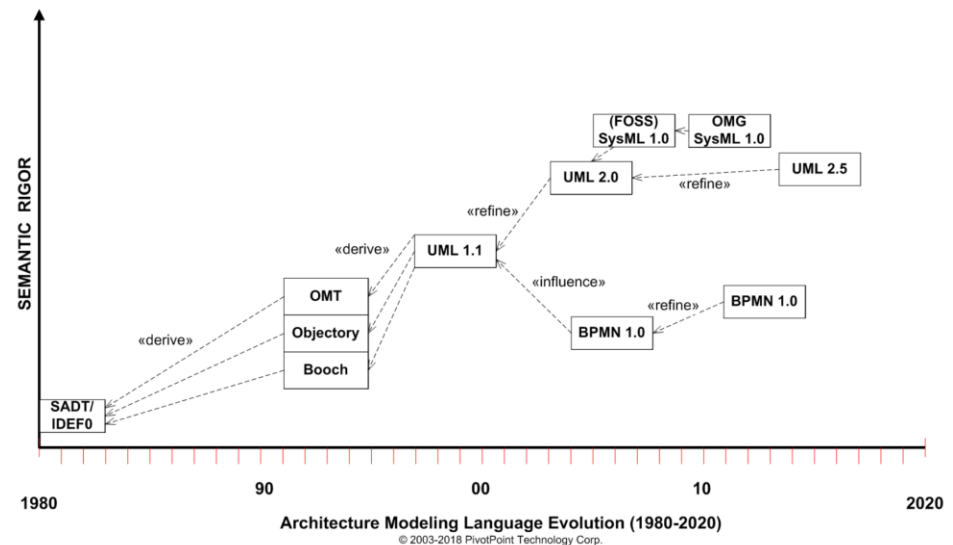
UML

# Versões da UML

Versão	Data de Lançamento
2.5	Junho de 2015
2.4.1	Agosto de 2011
2.4	Março de 2011
2.3	Maio de 2010
2.2	Fevereiro de 2009
2.1.2	Novembro de 2007
2.1.1	Agosto de 2007
2.0	Julho de 2005
1.5	Março de 2003
1.4	Setembro de 2001
1.3	Março de 2000
1.1	Novembro de 1997

# Evolução da UML

- Principais mudanças 1.1 para 2.0:
  - Novos diagramas:
    - Diagrama de objetos;
    - Diagrama de pacotes;
    - Diagrama de estruturas compostas
    - Diagrama de Visão geral de interação;
    - Diagrama de tempo;
    - Diagrama de perfil;
  - Digrama de colaboração foi renomeado para comunicação;
  - Melhorias nos diagramas de atividade e sequência;



Fonte da figura: <https://umlforum.com/uml-specifications/>

# O que é a UML (*Unified Modeling Language*)?

- Objetivos
  - Fornecer ferramentas aos engenheiros de software para **analisar, projetar e implementar sistemas baseados em software**, além da modelagem de processos de negócio e similares;
  - Avançar o estado da indústria, habilitando a **interoperabilidade entre ferramentas de modelagem visual de objetos**.
- A UML fornece:
  - Uma definição formal de um **metamodelo comum que especifica a sintaxe abstrata da UML**.
    - A sintaxe abstrata define o **conjunto de conceitos de modelagem UML**, seus **atributos** e seus **relacionamentos**, bem como as **regras para combinar esses conceitos para construir modelos UML** parciais ou completos.
  - Uma explicação detalhada da **semântica de cada conceito de modelagem UML**.
    - A semântica define, em uma maneira independente de tecnologia, **como os conceitos UML devem ser entendidos por computadores**.
  - Uma **especificação dos elementos de notação legíveis por humanos** para **representar os conceitos individuais da modelagem UML**, bem como **regras para combiná-los** em uma variedade de **diferentes tipos de diagramas** correspondentes a diferentes aspectos de sistemas modelados.
- De maneira geral, linguagem = vocabulário + regras de combinação (sintaxe)

# Modelos

- O que é um modelo?
  - Um modelo é uma **simplificação** (representação) da realidade.
  - Por que simplificação?
    - A **realidade** é muito **complexa** para ser representada em um único modelo.
- O que modelamos?
  - **Dimensões**: dados, função, comportamento, relacionamentos, etc.

# Objetivos da Modelagem

- **Documentar e compreender o problema;**
- **Projetar/criar a solução** para o problema;
- **Compreender** o software (solução) em desenvolvimento;
- Proporcionar uma **visão geral** do software;
- **Documentar decisões** tomadas durante o processo de desenvolvimento de software;
- Documentar a solução adotada utilizando **diferentes perspectivas** – todo software deve ser documentado através de um **conjunto de modelos;**
- **Especificar comportamento** ou a **estrutura** de um sistema.



# Princípios da Modelagem

- A escolha dos **modelos** a serem criados tem **profunda influência** sobre a maneira como um determinado **problema é atacado** e como uma **solução é definida**;
- Cada modelo pode ser expresso em diferentes **níveis de precisão** ;
- Os **melhores modelos** estão relacionados à **realidade** ;
- **Nenhum modelo único é suficiente**
  - Qualquer modelo não-trivial será melhor investigado por meio de um pequeno conjunto de **modelos relacionados, mas não redundantes**.

# A UML não é um

- um **processo**;
- um **método**;
- **análise e Projeto OO**;
- **regras de projeto**.

# Elementos da UML

- Para formar um **modelo conceitual** da linguagem é necessário aprender três elementos principais
  - **Blocos de construção;**
  - **Regras** que determinam como esses blocos poderão ser combinados;
  - **Mecanismos** comuns aplicados na UML.

# Blocos de Construção

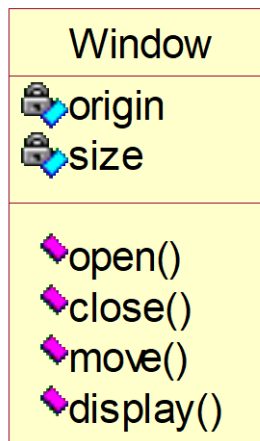
- Três tipos:
  - **Itens:** são abstrações;
  - **Relacionamentos:** os relacionamentos reúnem esses itens;
  - **Diagramas:** agrupam coleções interessantes deste item.

# Itens da UML

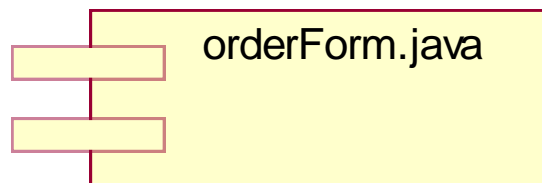
- Estruturais;
- Comportamentais;
- De agrupamento;
- Anotacionais.

# Itens estruturais

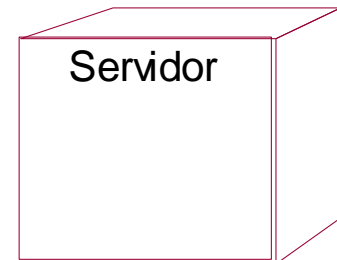
- São os substantivos dos modelos. São a **parte estática**, representando elementos **conceituais** ou **físicos**
- Sete tipos: **classes**, **interfaces**, **colaborações**, **casos de uso**, **classes ativas**, **componentes** e **nós**



**Classe**



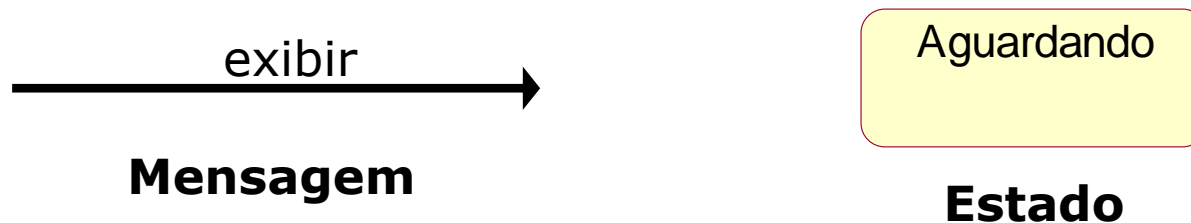
**Componente**



**Nó**

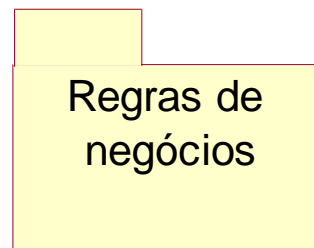
# Itens comportamentais

- Representam as partes **dinâmicas** dos modelos. São os verbos, representando **comportamentos** no tempo e no espaço
- Dois tipos: **interação** e **máquina de estado**



# Itens de agrupamento

- São as partes **organizacionais** dos modelos de UML. São os blocos em que os modelos podem ser decompostos – pacotes
- Um **pacote** é um mecanismo de **propósito geral** para a organização de elementos em grupos

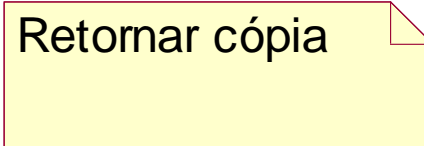


**Pacote**



# Itens anotacionais

- Partes **explicativas dos modelos UML**. São comentários, incluídos para descrever, esclarecer e fazer alguma observação importante sobre qualquer elemento do modelo - notas



Retornar cópia

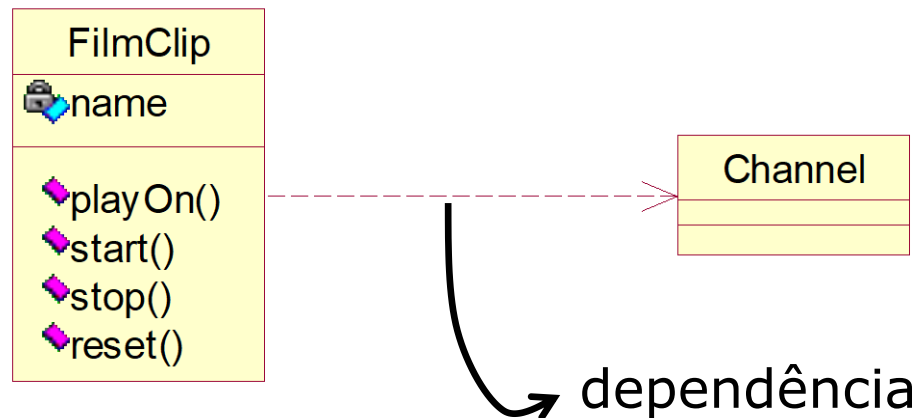
Nota

# Relacionamentos

- Dependência;
- Associação;
- Generalização;
- Realização;
- Composição;

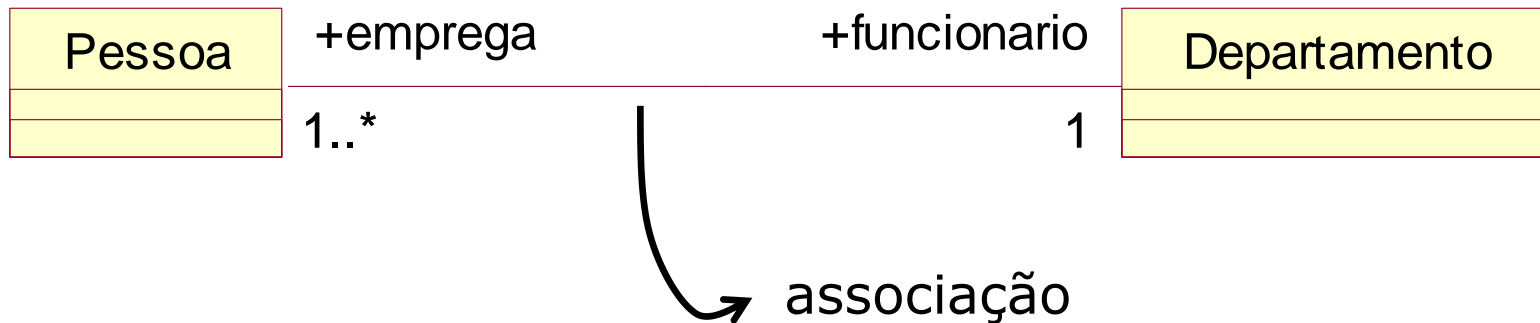
# Dependência

- Relacionamento **semântico** entre dois itens, nos quais a **alteração de um** (o item independente) **pode afetar** a semântica do **outro** (o item dependente)



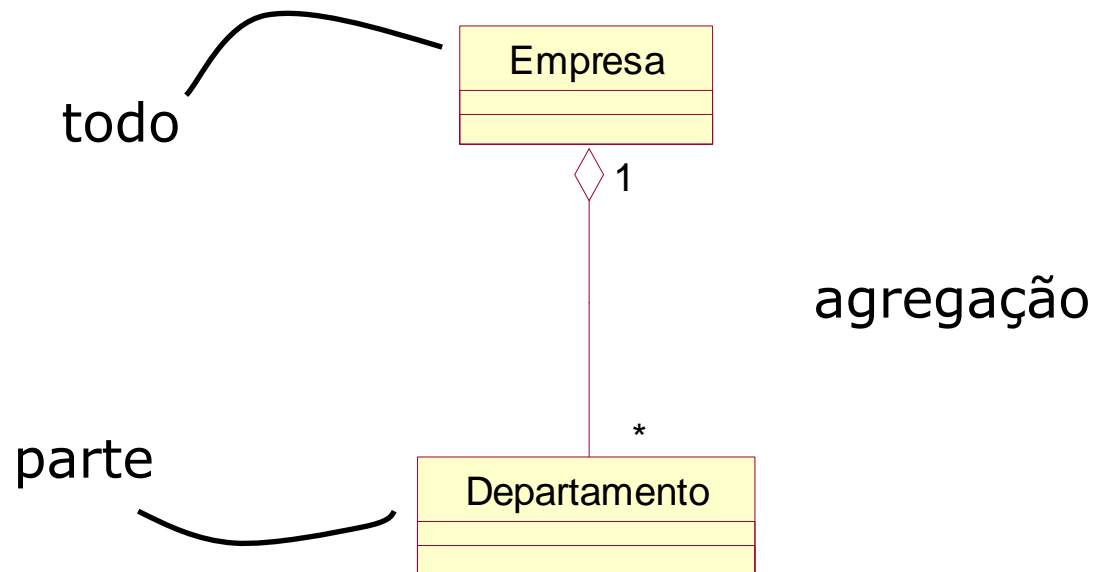
# Associação

- É um **relacionamento estrutural** que descreve um **conjunto de ligações**, em que as ligações são **conexões** entre objetos



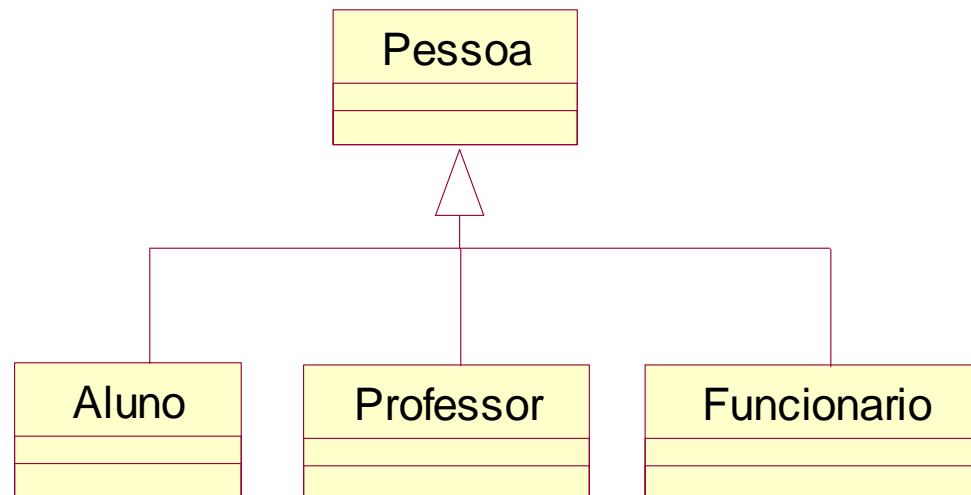
# Agregação

- A agregação é um **tipo especial de associação** representando um relacionamento estrutural entre o **todo e sua parte**



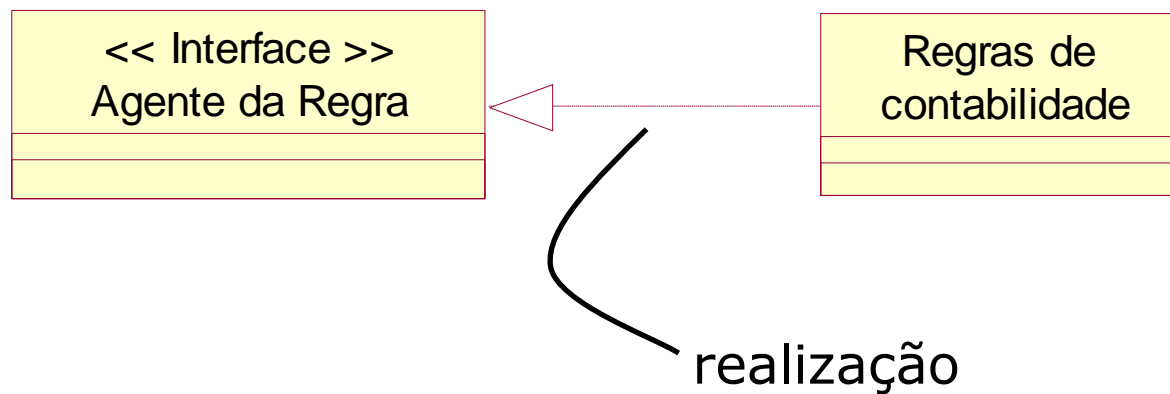
# Generalização

- É um **relacionamento de especialização/generalização**, nos quais os objetos dos **elementos especializados** (os filhos) são **substituíveis** por objetos do elemento generalizado (os pais)



# Realização

- É um **relacionamento semântico** entre **classificadores**, em que um classificador **especifica um contrato** que outro classificador garante executar



# Diagramas

- **Apresentações gráficas** de um conjunto de elementos, geralmente representadas como **gráficos de vértices** (itens) e **arcos** (relacionamentos)
- Nove tipos: **classes, objetos, pacotes, casos de uso, sequências, colaborações, estados, atividades, componentes e implantação**
- Podem ser classificados como de **estrutura, comportamento e interação**



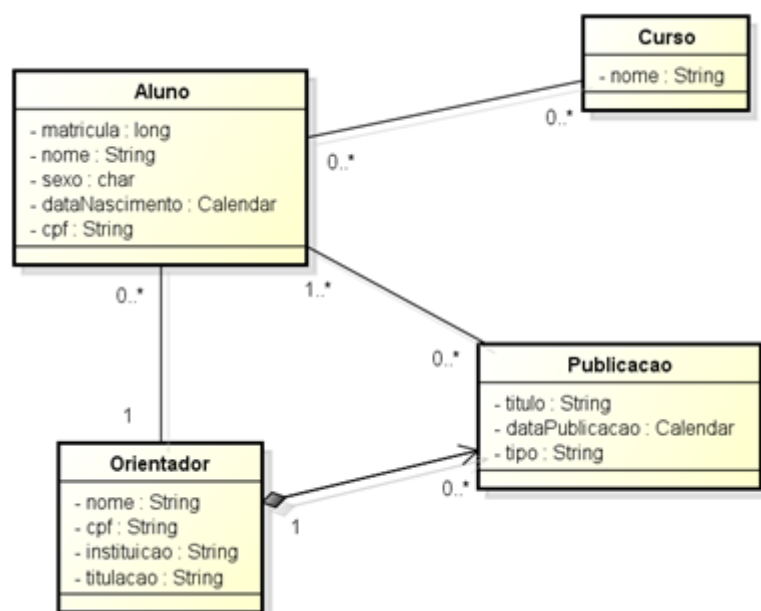
# Diagramas

- **Diagramas de estrutura**
  - **Enfatizam os elementos** que devem estar presentes no sistema modelado.
  - Como representam a estrutura, são muito utilizados para documentar a **arquitetura de software**.
- **Diagramas de comportamento**
  - Enfatizam o que **deve acontecer no sistema** que está sendo modelado.
  - Como ilustram o comportamento de um software, são muito utilizados para **descrever as funcionalidades**.
- **Diagramas de interação**
  - **Subconjunto de diagramas de comportamento**.
  - Enfatizam o **fluxo de controle e dados** entre os elementos do software.

# Diagramas UML

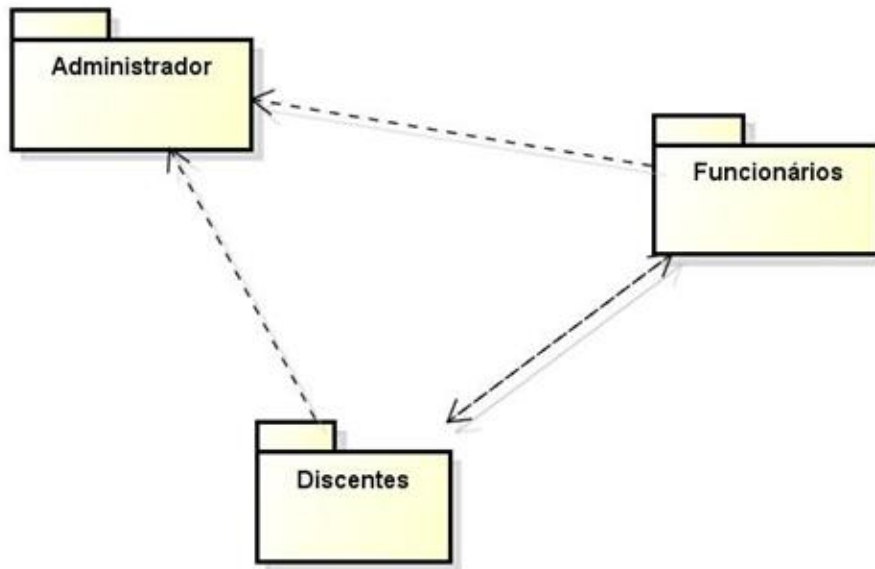
- Diagramas UML
  - Diagramas de estrutura
    - Diagrama de perfil
    - **Diagrama de classes**
    - Diagrama de estruturas compostas
    - **Diagrama de componentes**
    - **Diagrama de implantação**
    - **Diagrama de objetos**
    - **Diagrama de pacotes**
  - Diagramas de comportamento
    - **Diagrama de atividades**
    - **Diagrama de casos de uso**
    - **Diagrama de máquinas de estado**
    - Diagramas de interação
      - **Diagrama de sequência**
      - **Diagrama de comunicação**
      - Diagrama de visão geral de interação
      - Diagrama de tempo

# Diagrama de Classes



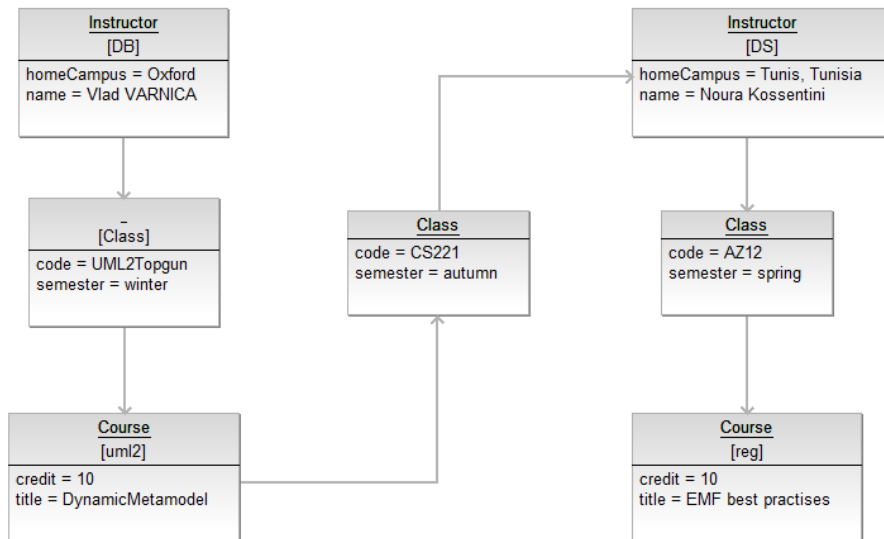
- Descrevem a **estrutura estática do sistema**, exibindo as **classes**, seus **atributos**, **operações** e **relacionamentos**.

# Diagrama de Pacotes



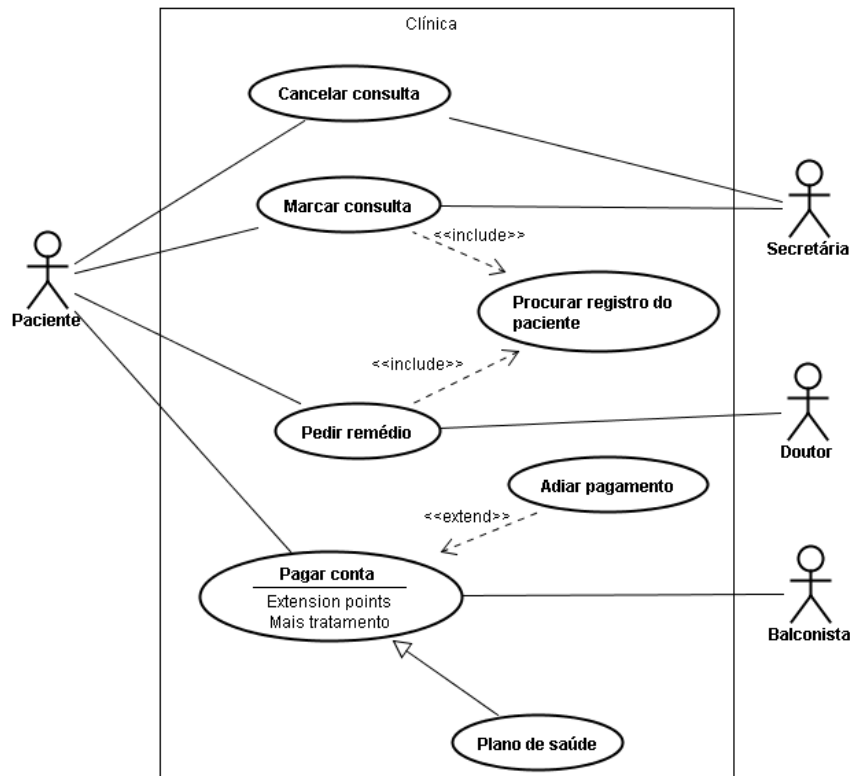
- Organizam elementos do sistema em grupos relacionados a fim de minimizar a dependência entre eles.
- Descrevem as dependências entre os pacotes do sistema.

# Diagrama de Objetos



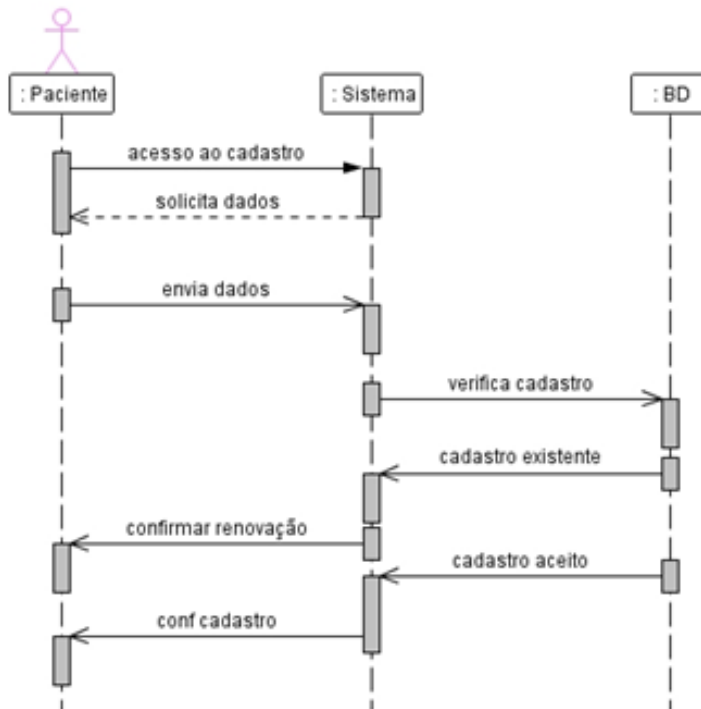
- Descrevem a estrutura estática de um sistema em um determinado momento
- Podem ser usados para testar a precisão dos diagramas de classe

# Diagrama de Casos de Uso



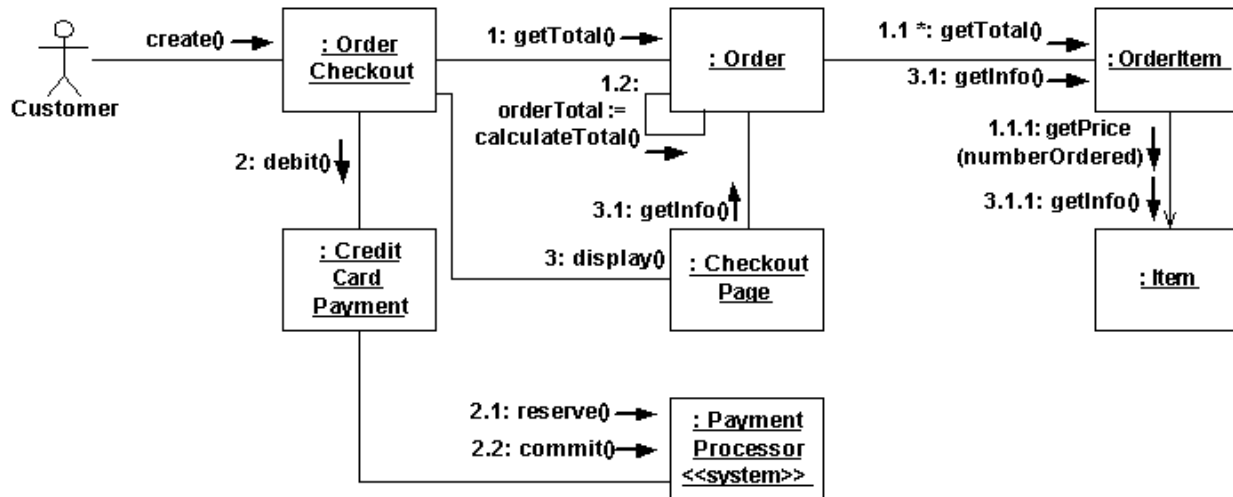
- Modelam a funcionalidade do sistema através de atores e casos de uso
- Casos de uso são funcionalidades fornecidas pelo sistema aos seus usuários

# Diagrama de Sequência



- Descreve as interações entre as classes através das trocas de mensagens ao longo do tempo

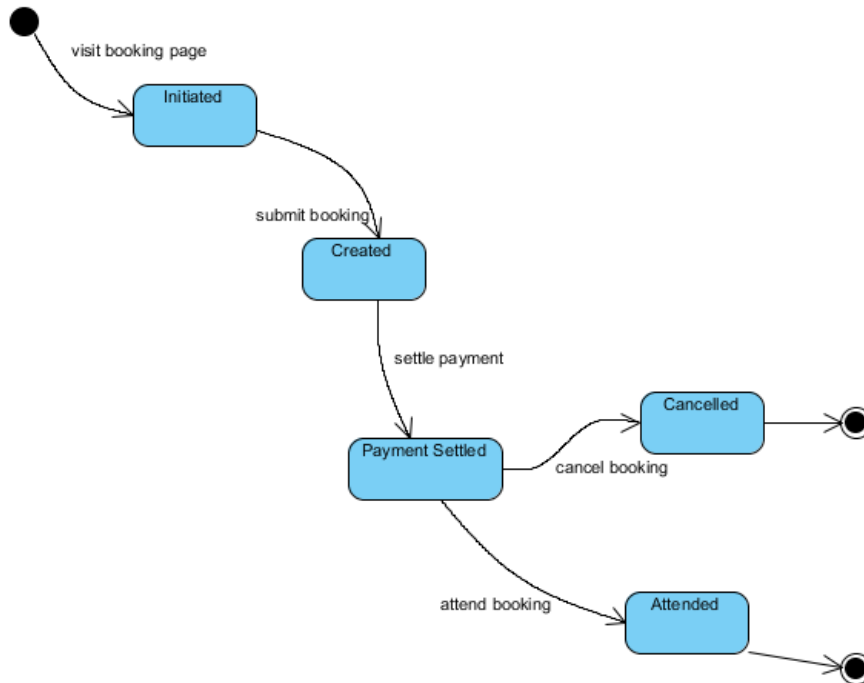
# Diagrama de Comunicação



- Representam as interações entre objetos em termos de mensagens em sequência
- Descrevem tanto a estrutura estática como o comportamento dinâmico do sistema

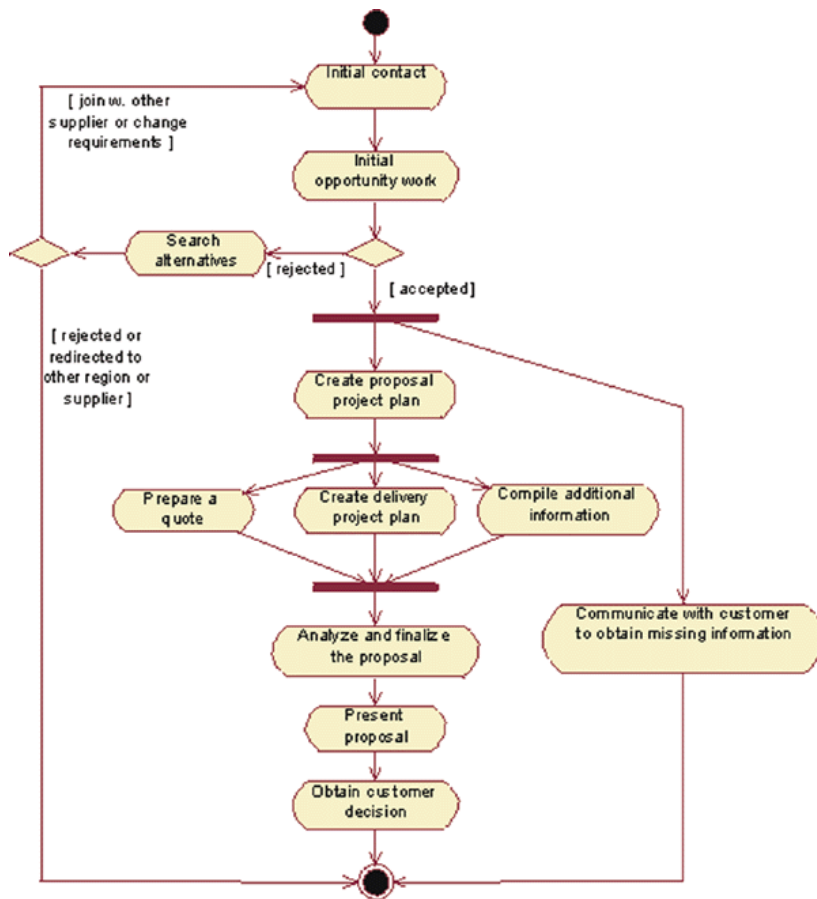


# Diagrama de Estados



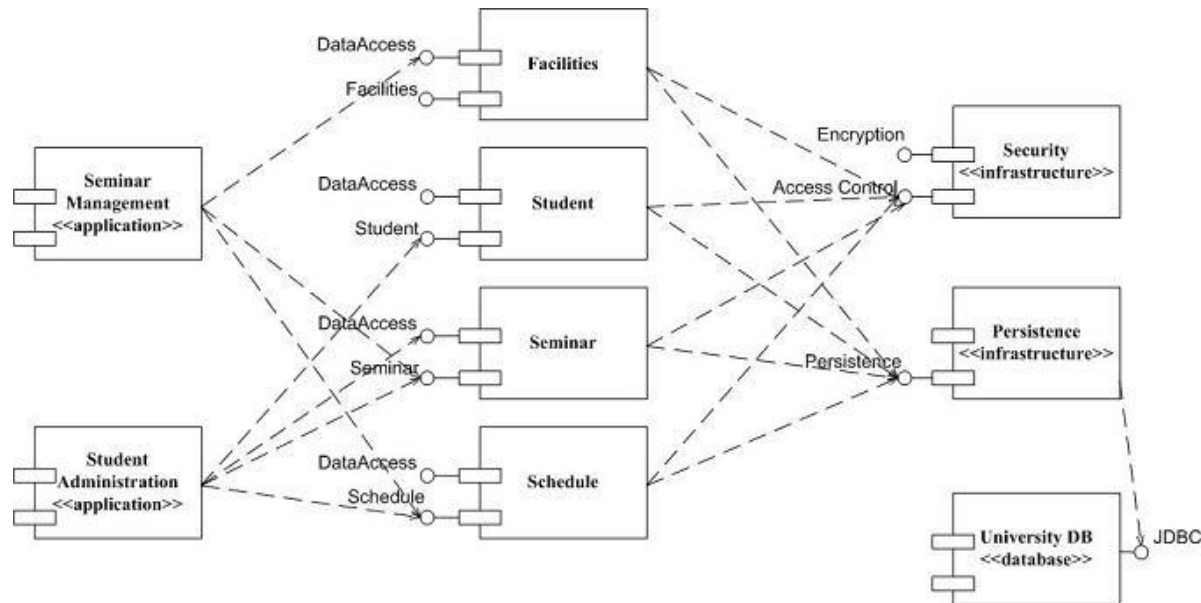
- Descrevem o comportamento dinâmico do sistema em resposta a estímulos externos
- São especialmente úteis para modelar objetos reativos cujos estados são disparados por eventos específicos

# Diagrama de Atividades



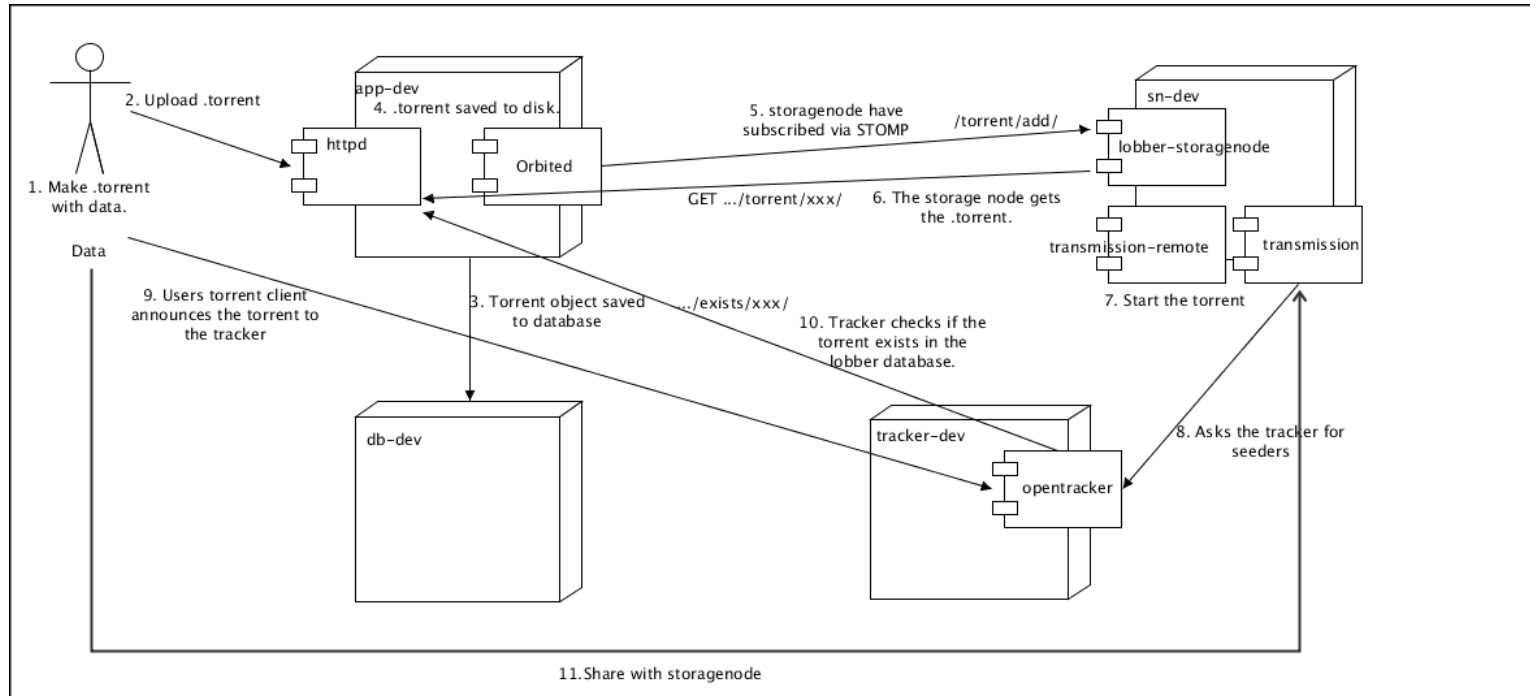
- Ilustram a natureza dinâmica de um sistema modelando o fluxo de controle de uma atividade para outra.
- Uma atividade representa uma operação em uma classe do sistema que resulta na mudança do estado do sistema.
- Tipicamente, são usados para modelar fluxo de trabalho ou processos de negócio e funcionamento interno.

# Diagrama de Componente



- Descreve a organização dos componentes físicos de software
- Ex.: código-fonte, código em tempo de execução (binário) e executáveis

# Diagrama de Implantação



- Descrevem os recursos físicos em um sistema, incluindo nós, componentes e conexões

# Regras UML

- Especificam o que deverá ser um modelo bem-formado;
- Modelos bem-formados são aqueles auto consistentes semanticamente e em harmonia com todos os modelos a ele relacionados;
- Regras para: nome, escopo, visibilidade, integridade e execução.

# Material de apoio

- Bibliografia básica

- BOOCH, G., RUMBAUGH, J., JACOBSON, Ivar. Uml - Guia do Usuário. 2006. Editora GEN LTC.

- Bibliografia complementar

- Booch, Grady, Ivar Jacobson, and James Rumbaugh. "The unified modeling language." Unix Review 14.13 (1996).



INSTITUTO DE CIÊNCIAS EXATAS

DEPARTAMENTO DE COMPUTAÇÃO

Perguntas?