

47006- ANÁLISE E MODELAÇÃO DE SISTEMAS

Modeling the domain concepts with classes

Ilídio Oliveira

v2020/11/04, TP09

Learning objectives for this lecture

- Justify the use of structural models in systems specification.
- Draw a simple class diagram to capture the concepts of a problem domain.
- Describe the types and roles of the different associations in the class diagram.
- Explain the relationship between class and objects; and between class and object diagrams.

Objetos: classificar as “coisas” do mundo

Desenvolvimento por objetos

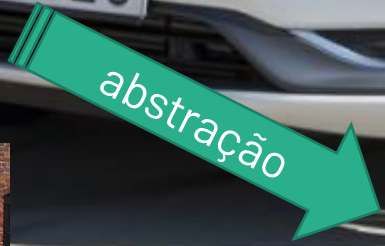
É uma estratégia para simplificar o espaço do problema, modularizando-o

Esquema mental comum à análise (requisitos) e programação

Classificar entidades similares

Facilita a reutilização de software

Orientado pelo "vocabulário" do domínio



abstração

Veiculo
-matricula
-cor
-lotação
-velocidade atual
+travar()
+acelerar()

3 mecanismos conceptuais para modelar/gerir a complexidade

Abstração



- Tratar os objetos que são semelhantes como um "tipo"/categoria

Encapsulamento



- O objeto é uma unidade que esconde o seu estado interno
- A interação com o objeto é feita através de "pontos de acesso"

Hierarquia



- Procurar relações "é-um"
- Procurar relações "parte-de"

Mecanismo: abstração

abstração decorre do reconhecimento de semelhanças entre certos objetos, situações ou processos no mundo real, e a decisão de se concentrar nestas semelhanças → um *tipo de coisa*

Uma abstração denota as características essenciais de um objeto que o distingue de todos os outros tipos de objetos.

Abstraction: Temperature Sensor
Important Characteristics: temperature location
Responsibilities: report current temperature calibrate

Figure 2–6 Abstraction of a Temperature Sensor

Abstraction: Heater
Important Characteristics: location status
Responsibilities: turn on turn off provide status

Related Candidate Abstractions: Heater Controller, Temperature Sensor

Figure 2–9 Abstraction of a Heater

Como é que o encapsulamento contribui para a gerir a complexidade?

Encapsulamento ajuda a gerir a complexidade escondendo a dimensão “privada” (interna) das nossas abstrações.

Abstraction:	Temperature Sensor
Important Characteristics:	temperature location
Responsibilities:	report current temperature calibrate

Figure 2–6 Abstraction of a Temperature Sensor

Abstraction:	Heater
Important Characteristics:	location status
Responsibilities:	turn on turn off provide status

Related Candidate Abstractions: Heater Controller, Temperature Sensor

Figure 2–9 Abstraction of a Heater

O que é que *Heater* deve conhecer de *Temperature Sensor*?

Hierarquia: a relação de herança

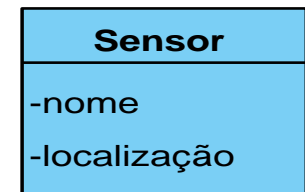
A herança define uma relação entre classes (tipos de coisas), em que uma classe partilha/especializa a estrutura ou o comportamento definido numa outra classe.

A herança denota uma relação semântica "é-um", e.g.:

um urso "é um" (tipo de) mamífero;
uma casa "é um" tipo de ativo tangível; uma caravana é um veículo.

A herança implica, assim, uma generalização \leftrightarrow especialização

Visual Paradigm Standard (I Oliveira) (Universidade de Aveiro)



Abstraction:	Temperature Sensor
Important Characteristics:	temperature location
Responsibilities:	report current temperature calibrate

Figure 2-6 Abstraction of a Temperature Sensor

Qual a relação semântica?
Um Sensor de temperatura
é um (**is-a**) Sensor.

Hierarquia: a relação de agregação

A agregação define uma relação entre categorias de coisas (i.e. classes) do tipo “parte-de”

A agregação denota a relação semântica “parte-de”, e.g.:

Um País é parte de um Continente; o Estudante é parte da Turma; uma Obra é parte de uma Coleção.

A herança implica, assim, um agregador $\leftarrow \rightarrow$ parte-de

Abstraction: Heater
Important Characteristics: location status
Responsibilities: turn on turn off provide status

Related Candidate Abstractions: Heater Controller, Temperature Sensor

Figure 2-9 Abstraction of a Heater

Abstraction: Temperature Sensor
Important Characteristics: temperature location
Responsibilities: report current temperature calibrate

Figure 2-6 Abstraction of a Temperature Sensor

O *Temperature Sensor* é parte de um *Heater*.

Um Estudante é parte de uma Turma.

3 mecanismos conceptuais para modelar/gerir a complexidade

Abstração



- Tratar os objetos que são semelhantes como um "tipo"/categoria

Encapsulamento



- O objeto é uma unidade que esconde o seu estado interno
- A interação com o objeto é feita através de "pontos de acesso"

Hierarquia



- Procurar relações "é-um"
- Procurar relações "parte-de"

Classes and objects

In object-oriented modeling

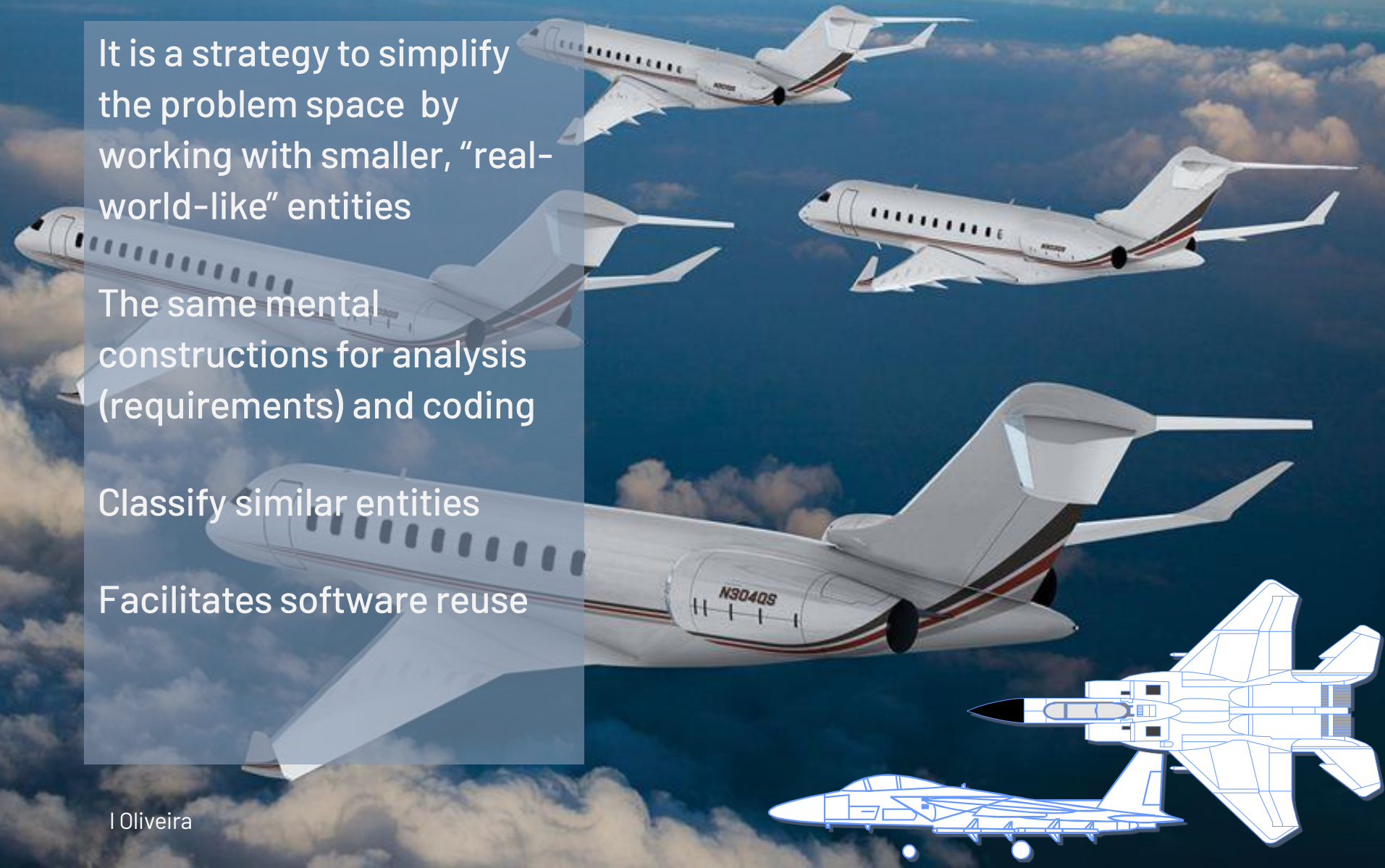
Object-oriented development

It is a strategy to simplify the problem space by working with smaller, “real-world-like” entities

The same mental constructions for analysis (requirements) and coding

Classify similar entities

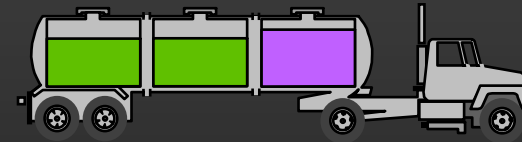
Facilitates software reuse



Objects (in OO) model real-world/problem space entities

Observable in the physical world

e.g.: Student, airplane



Truck

Concepts

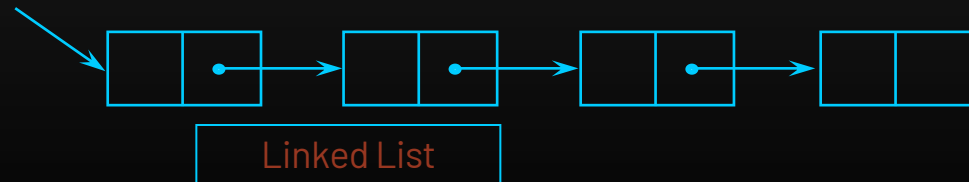
e.g.: sale, booking

Software abstractions

e.g.: LinkedList, Vector



Chemical Process



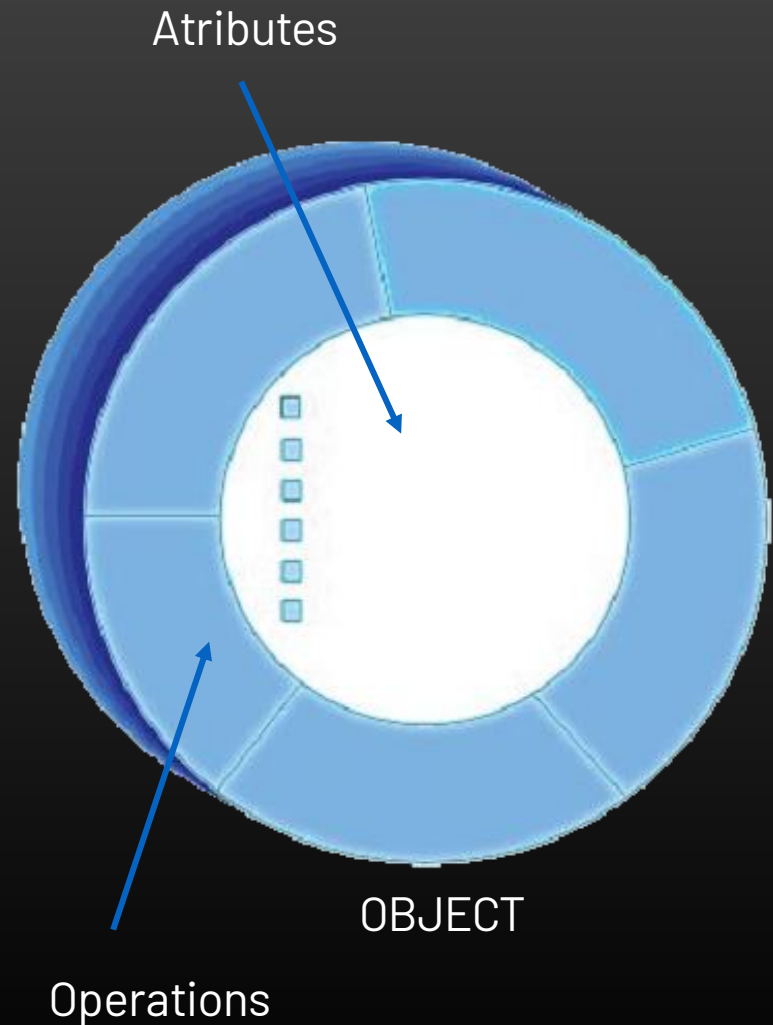
Object = purpose + state + behavior

An object has a well-defined purpose

It is an entity with a boundary that encapsulates state and behavior.

The state is represented through attributes and relationships.

The behaviorally is represented through operations.



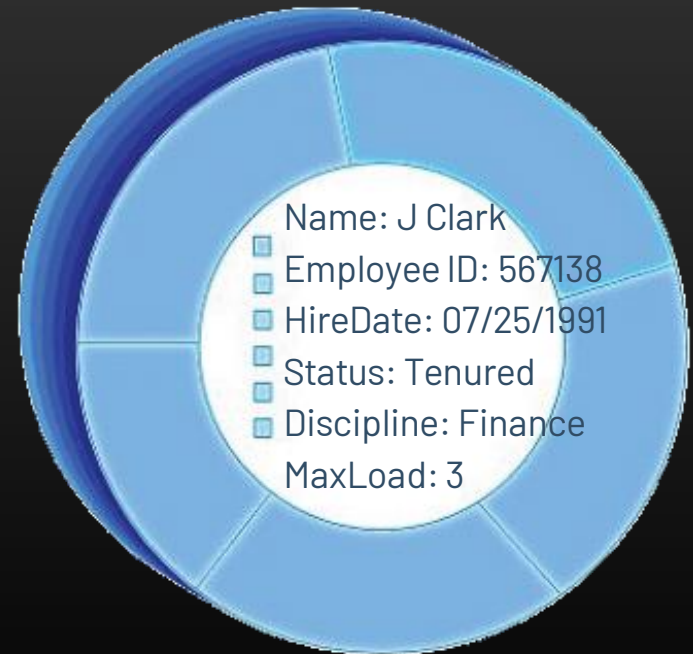
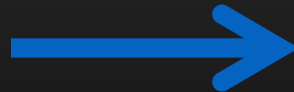
An object has a (internal) state

The state of an object matches one of the conditions/settings is that it is possible for the object to present itself.

It is normal for the object state to change over time.



Name: J Clark
Employee ID: 567138
Date Hired: July 25, 1991
Status: Tenured
Discipline: Finance
Maximum Course Load: 3 classes

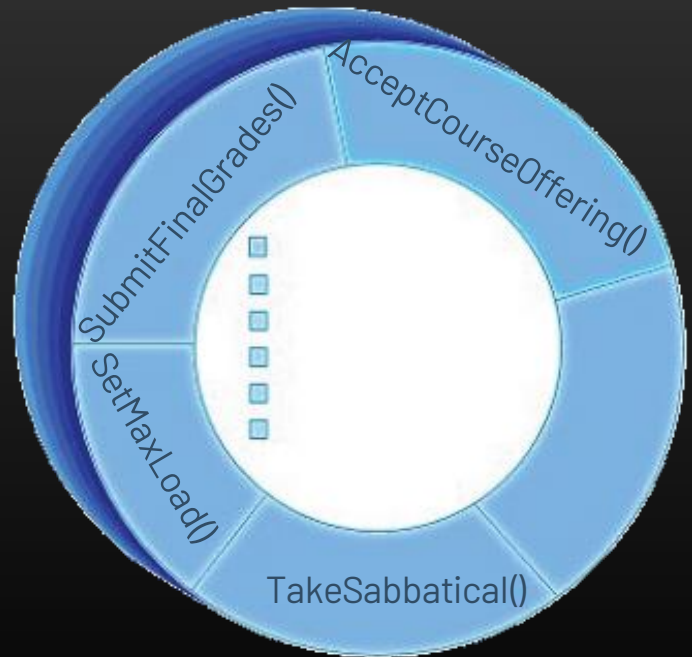
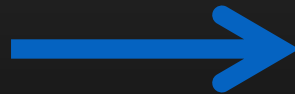


Professor Clark

An object has behavior (functionality)

The behavior defines how the object acts/reacts

The visible/exposed behavior is modeled by the set of messages that it responds to (operations)



Professor Clark's behavior
Submit Final Grades
Accept Course Offering
Take Sabbatical
Maximum Course Load: 3 classes

Professor Clark

Object: State + operations

You know something

Its attributes

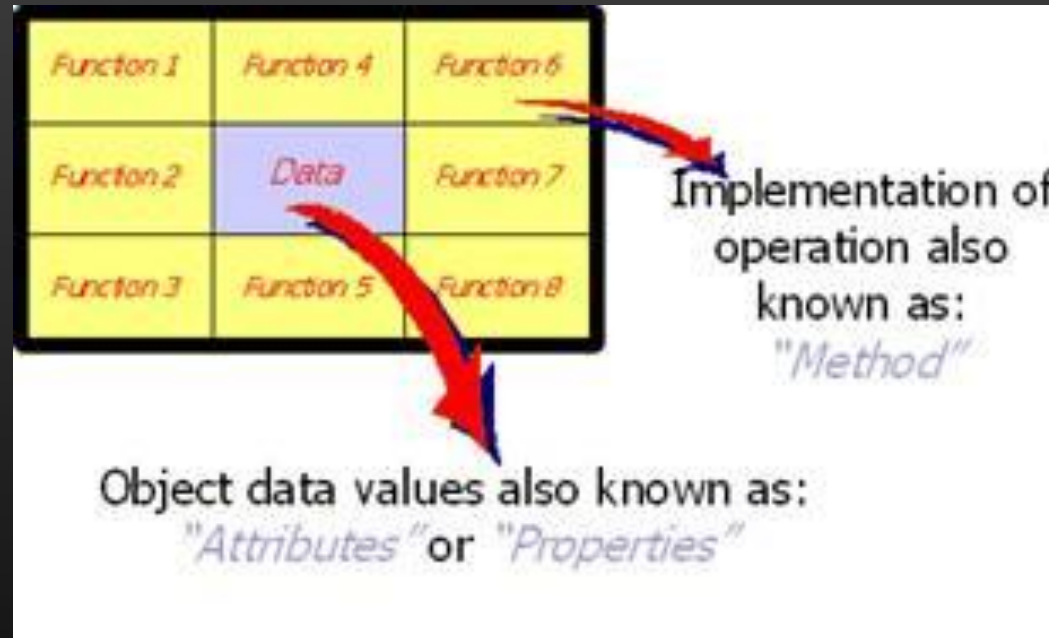
Your relationships

Do you know how to do something

Its behaviour, activated through operations

Capsule

The others (objects) do not need to know what he knows...



An object has its own identity

Each object has its own unique identity, even if its state is equal to that of another object.



Professor "J Clark"
teaches Biology



Professor "J Clark"
teaches Biology

What's a class?

It is a category of similar objects, which share the same attributes, operations, relationships, and semantics.

The object is an instance (occurrence) of a class

A class is an abstraction

Categorizes similar objects

Emphasizes the characteristics of interest (and suppresses others)



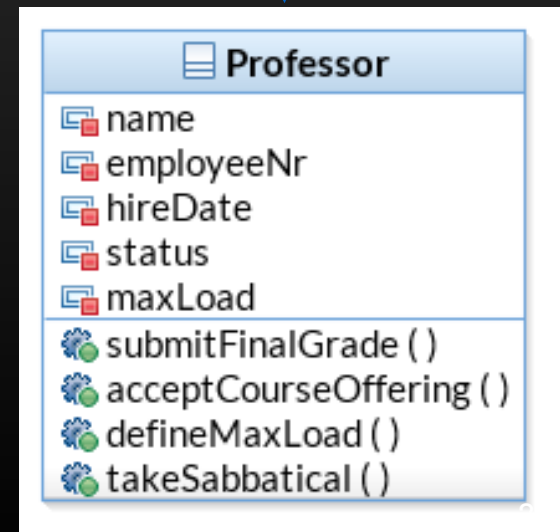
Professor Torpie



Professor Meijer



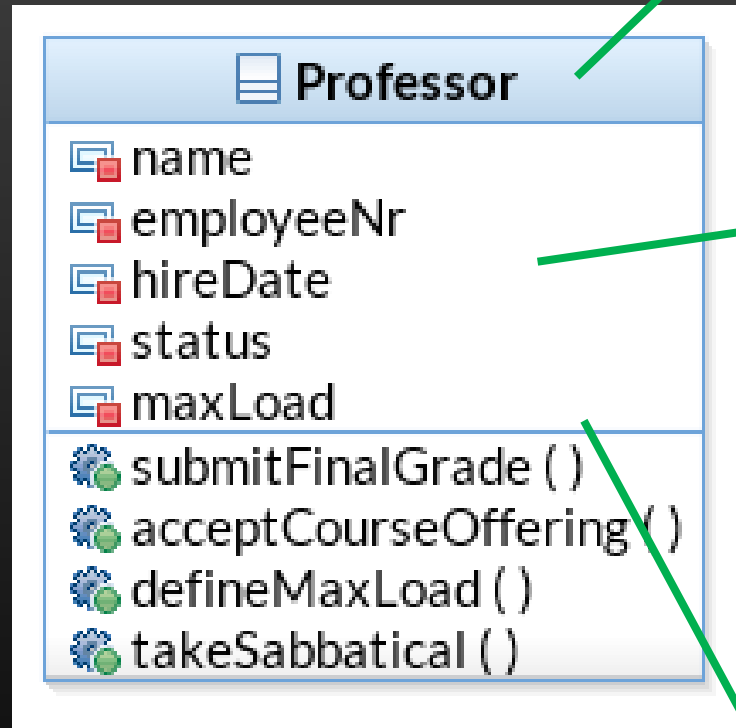
Professor Allen



UML class representation



Professor J Clark
I Oliveira



Name

Attributes

Operations

The relationship between classes and objects

A class is an abstract definition of an object

Defines the structure and behavior of each object in that class/category.

Works as a template for creating objects.

Classes are not collections of objects



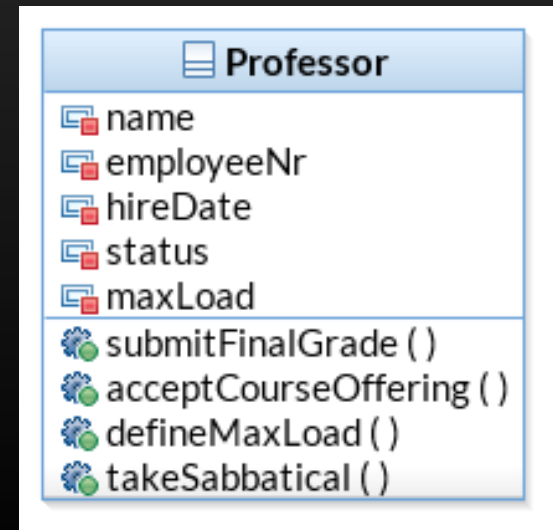
Professor Torpie



Professor Meijer



Professor Allen

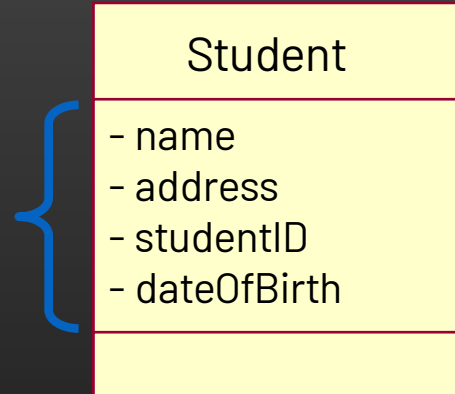


What is an attribute?

Is a property of a class that describes the sort of values that instances can hold

A class can have multiple or no attributes

Attributos



What's an operation?

Is the implementation of a service/action that can be requested from any object in a class

What the class can do

The class can have multiple or no operations

Common: commands and interrogations (lookups)

Operações

Student
+ getTuition() + addSchedule() + getSchedule() + deleteSchedule() + hasPrerequisites()

Arrivées

13:02

Compagnie	Vol	Provenance	Heure	Hall	Remarque
 AIR ALGERIE	AH 5017	Ouagadougou	05:40	2	Retardé
 QATAR	QR 1379	Doha	12:05	1	Arrivé 12:05
 IBERIA	IB 3308	Madrid	12:15	2	Arrivé 12:25
 AIR ALGERIE	AH 1023	Marseille	12:25	2	Arrivé 12:50
 AIGLE AZUR	ZI 707	Marseille	13:05	2	Retardé 13:30
 AIR ALGERIE	AH 1037	Lyon	13:10	2	Retardé 13:20
 Emirates	EK 757	Dubai	13:10	1	Atterri 13:00
 AIGLE AZUR	ZI 223	Paris-Orly	13:10	1	Retardé 13:20
 AIR ALGERIE	AH 1003	Paris-CDG	13:15	2	Retardé 14:40

11



CONFAC

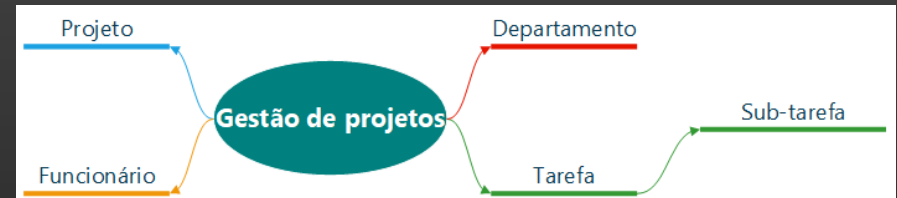
Os objetos de um domínio estão ligados numa “rede de conhecimento”

Um Projeto é coordenado por um determinado Departamento.

O Projeto tem uma equipa de vários Funcionário(s) atribuída.

Cada Projeto define várias Tarefas que, por sua vez, podem estar organizadas em sub-tarefas.

etc.



Como transportar o conhecimento do domínio pra um modelo?
Há “coisas” de interesse e “relacionamentos” entre elas

O que é a "associação"?

A relação semântica que se estabelece entre duas ou mais classes que descreve as ligações existentes entre as respectivas instâncias.

Mostra que objetos de um tipo estão ligados a objetos de outro tipo.

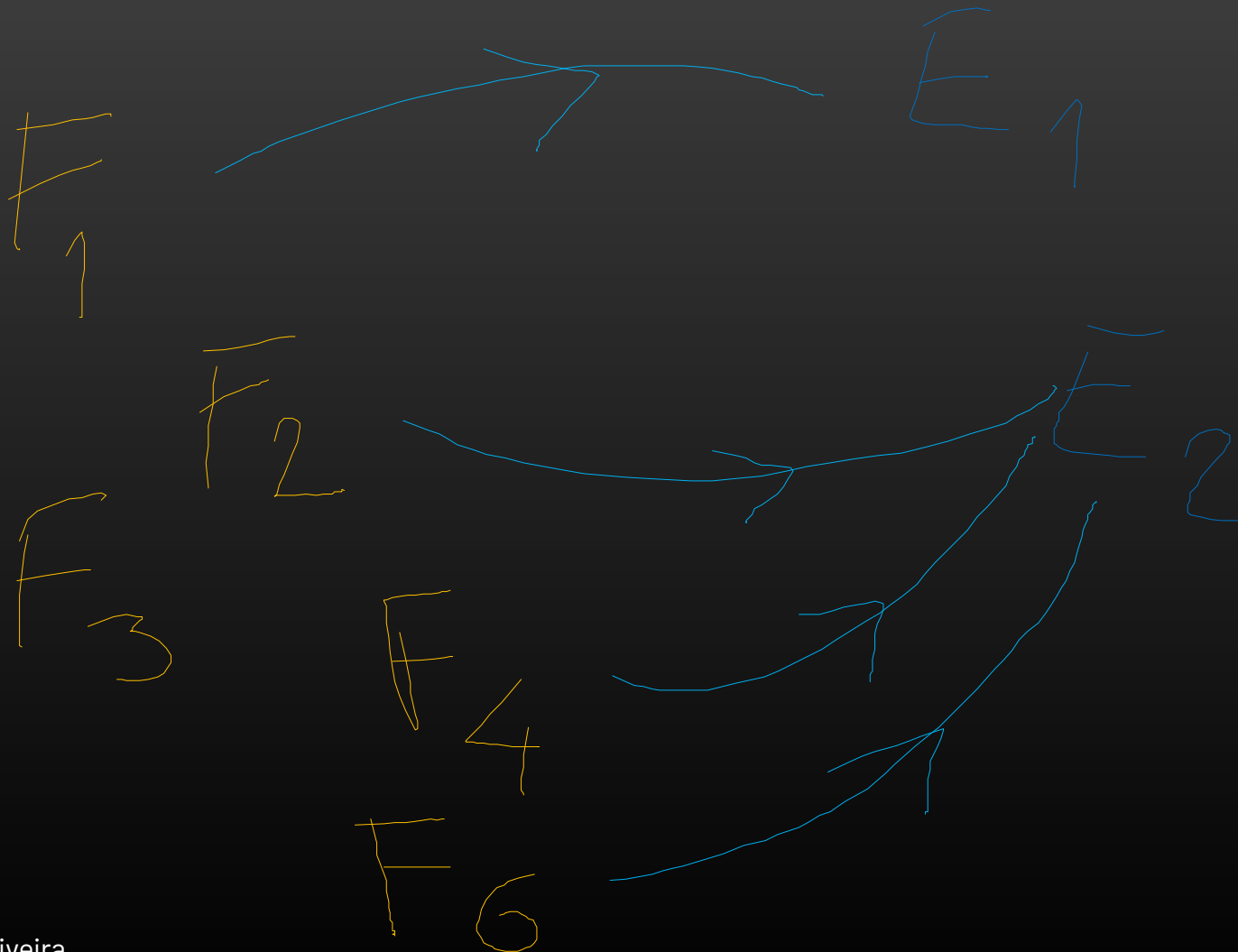
O tipo de "ligação" deve ser anotado com uma explicação do significado.

A classe Funcionário e Empresa estão associadas. A descrição da associação é "trabalha para".

Portanto:
"Funcionário" → "trabalha em" → "Empresa".



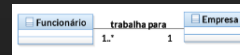
Com que “frequência” os objetos de F e E estão associados?



O que é a multiplicidade (de uma associação)?

Nr de instâncias de uma classe que se relacionam com uma instância da outra.


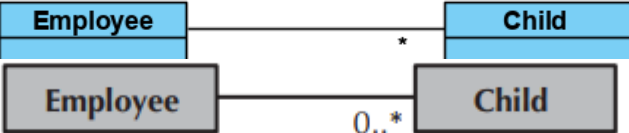



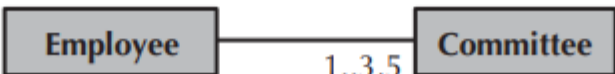
“O Funcionário só trabalha numa empresa; a Empresa tem pelo menos um Funcionário”



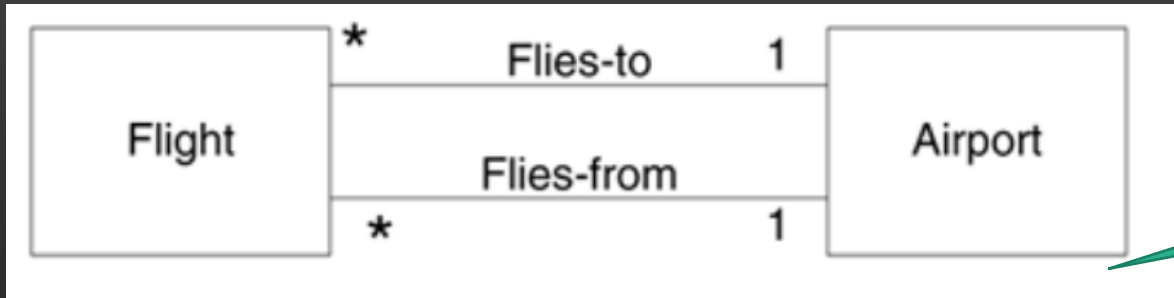
“O Funcionário só trabalha numa empresa; a Empresa só tem um Funcionário”



Indicação de multiplicidade

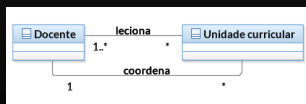
Exactly one	1		A department has one and only one boss.
Zero or more	0..*		An employee has zero to many children.
One or more	1..*		A boss is responsible for one or more employees.
Zero or one	0..1		An employee can be married to zero or one spouse.
Specified range	2..4		An employee can take from two to four vacations each year.
Multiple, disjoint ranges	1..3,5		An employee is a member of one to three or five committees.

Múltiplas associações entre duas classes



Um voo: parte de um Aeroporto; chega a um Aeroporto.
O conhecimento sobre um Voo associa 2 objetos Aeroporto, por razões diferentes

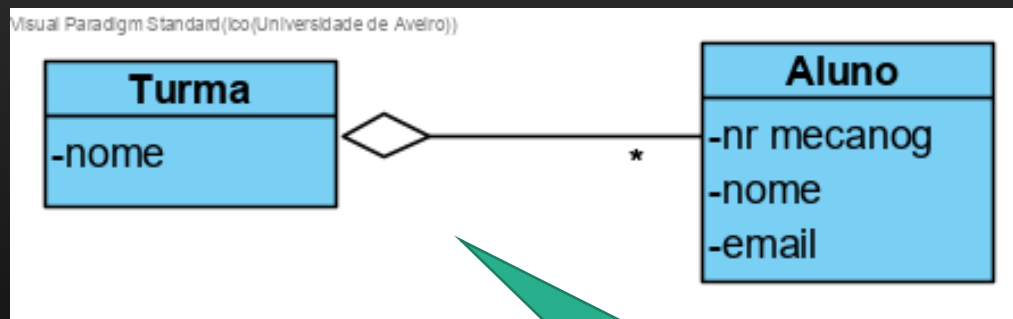
Um docente: leciona várias UC; coordena várias UC.



O que é uma agregação?

É uma forma especial de associação que modela uma relação de todo-parte, entre o agregador (“contentor”) e as suas partes constituintes

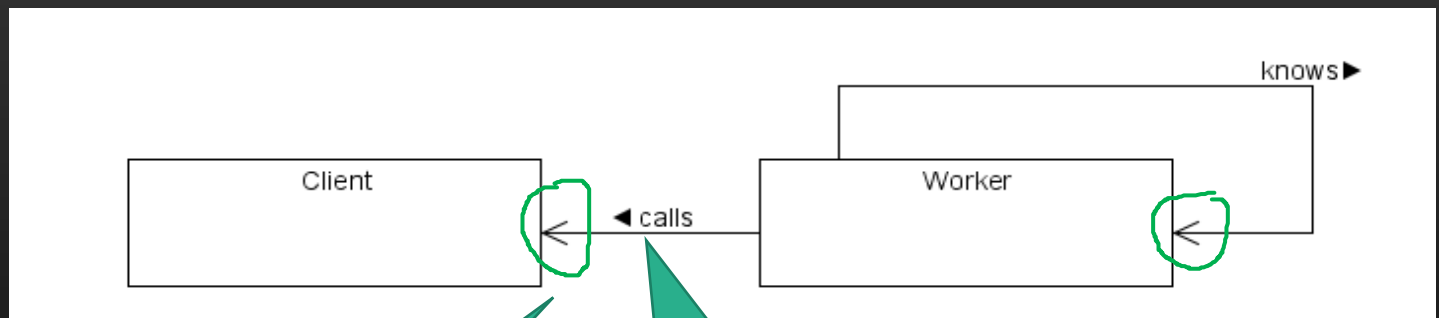
Deve ser natural ler-se “É parte de...” (sentido parte → todo); se for “forçado”, usar a associação normal



Um objeto *Turma* agrega vários objetos *Aluno*.

O que é a navegabilidade?

Indica a possibilidade de navegar de uma classe de partida para uma classe de chegada, usando a associação



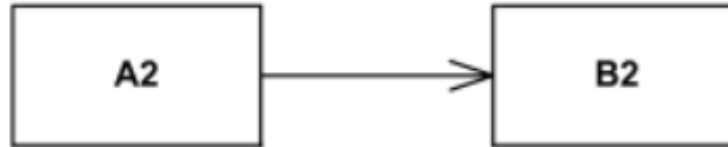
Em que sentido é que a associação é navegável/aplicável? (Se omissa → ambos)

Notação auxiliar: em que sentido se deve ler a etiqueta. (Worker > calls > Client.
(Se omissa: cima p/ baixo; esquerda p/ direita)

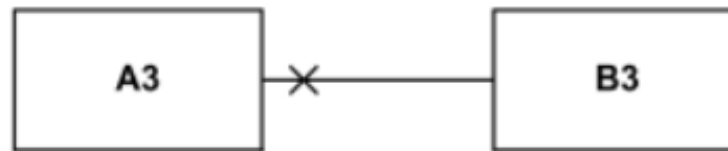
Os dois primeiros casos são as situações mais comuns.



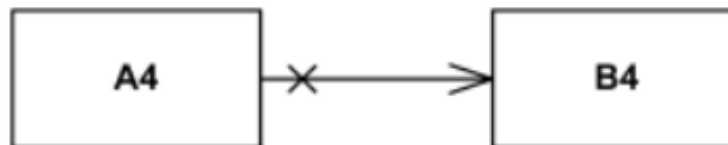
*Both ends of association have **unspecified navigability**.*



*A2 has **unspecified navigability** while B2 is **navigable** from A2.*



*A3 is **not navigable** from B3 while B3 has **unspecified navigability**.*



*A4 is **not navigable** from B4 while B4 is **navigable** from A4.*

<http://www.uml-diagrams.org/association.html?context=class-diagrams>

O que é a generalização (=herança)

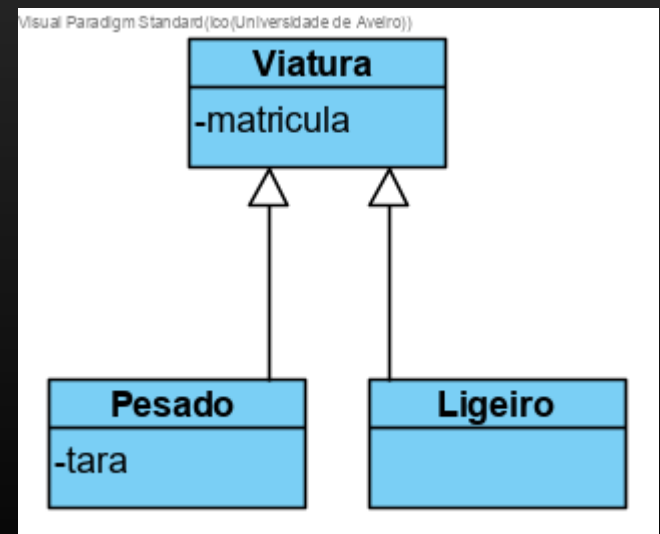
É relação entre classes em que uma especializa a estrutura e/ou comportamento de outra, partilhando todas as características

Define uma hierarquia em que a subclass herda das características da superclasse

A subclasse pode sempre ser usada onde a superclasse é usada, mas não ao contrário.

Pode ler-se “é um tipo de”

(Um *Pesado* é um tipo de *Viatura*, com matrícula e tara.)



O que é passado à subclasse?

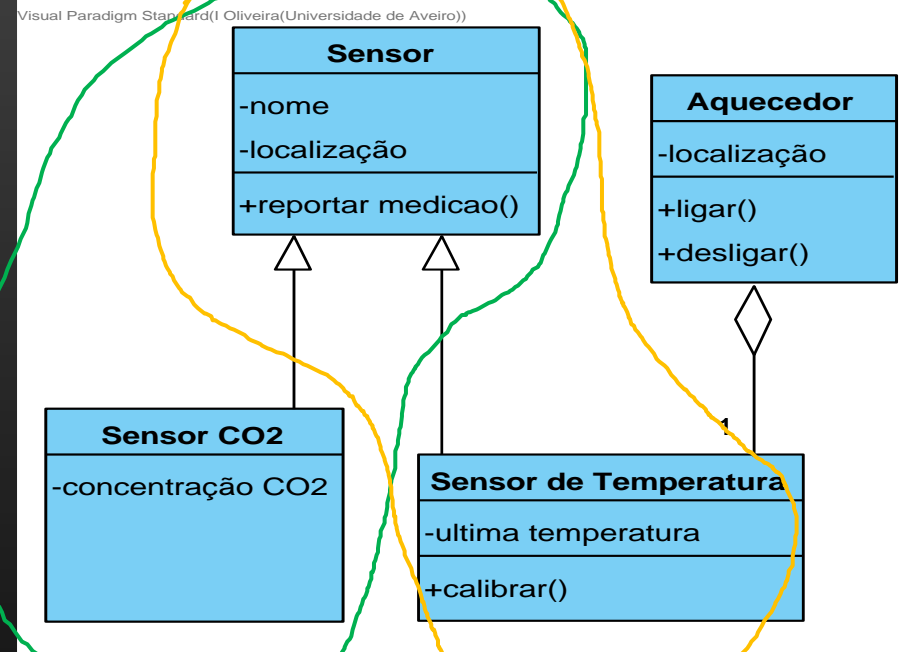
A subclasse herda os atributos, operações e relacionamentos da superclasse

A subclasse pode:

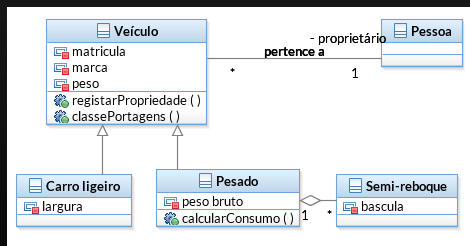
- Adicionar mais atributos, operações e relacionamentos à base herdada

- Redefinir as operações da superclasse

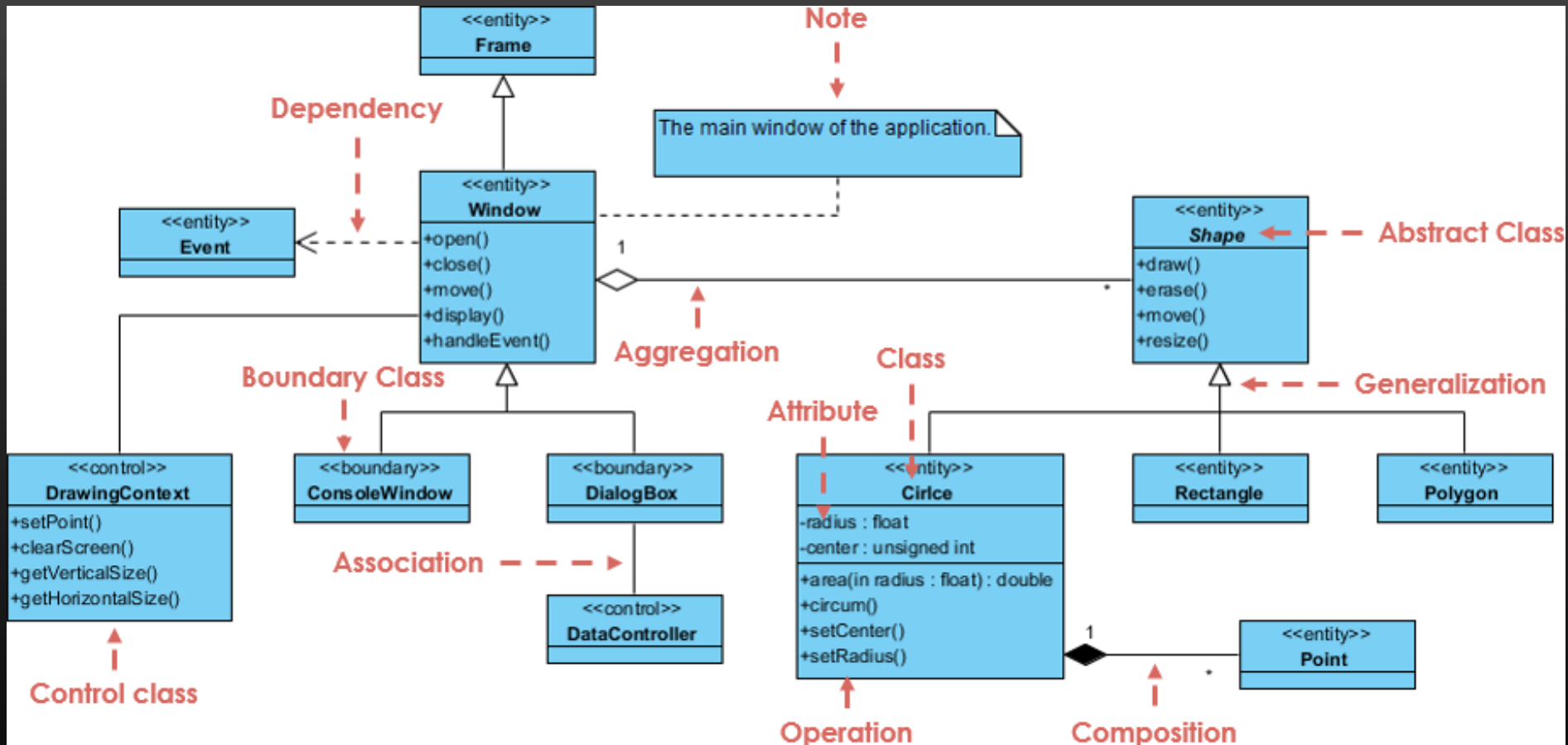
A herança põe em evidencia as características comuns entre classes



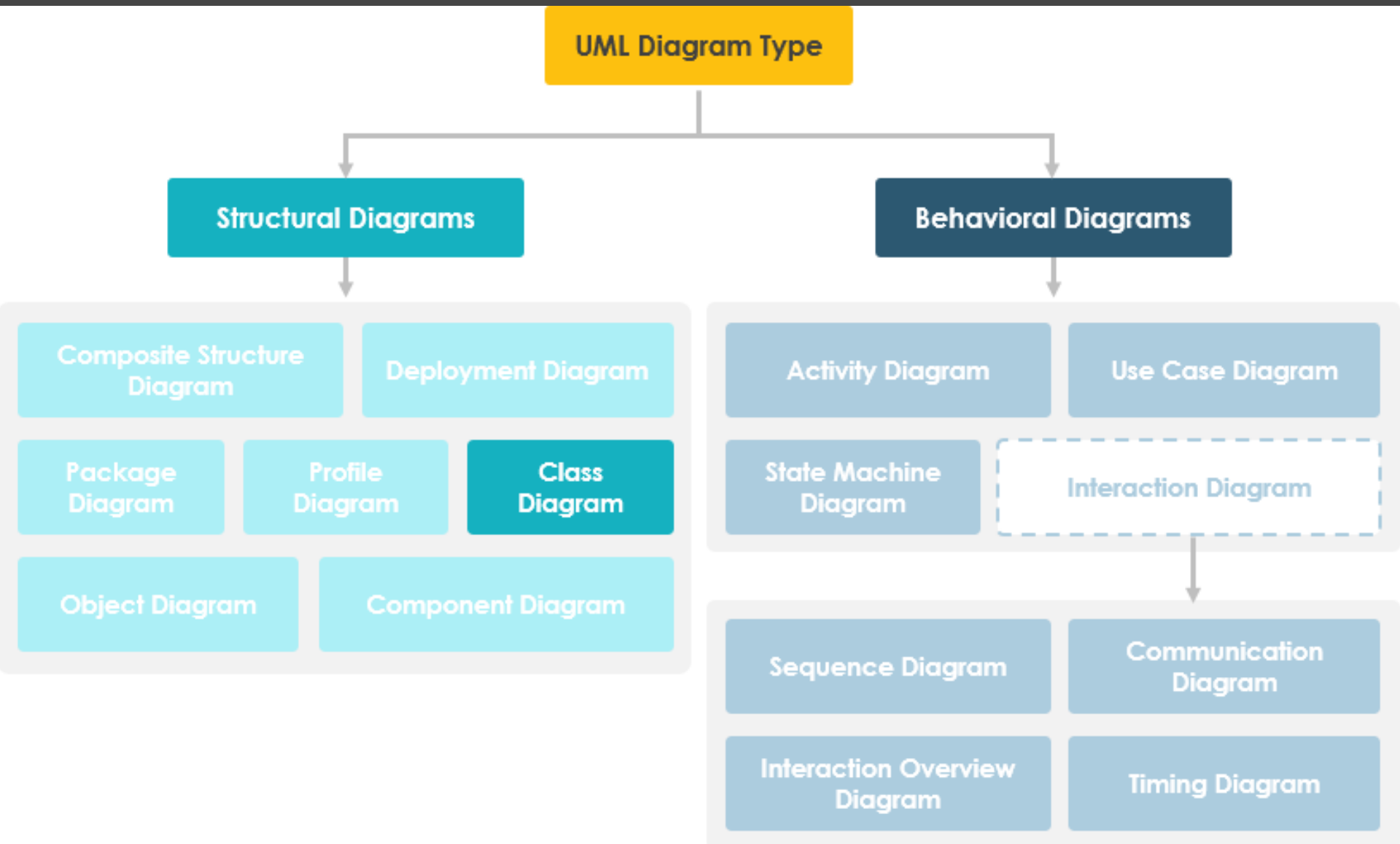
What is "inherited"?



Class Diagram notation overview



<https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-class-diagram/>



Exercício: domínio da gestão de projetos

Object-oriented “decomposition”

“Although both designs solve the same problem, they do so in quite different ways.

In this second decomposition, we view the world as a set of autonomous entities that collaborate to perform some higher-level behavior. *Get formatted update* thus does not exist as an independent algorithm; rather, it is an **operation associated with the object** *File of Updates*.

Calling this operation creates another object, *Update to Card*. In this manner, **each object in our solution embodies its own unique behavior, and each one models some object in the real world.**

Objects do things, and we **ask them to perform what they do by sending them messages.**

Because our decomposition is based upon objects and not algorithms, we call this an *object-oriented decomposition*.”



Readings & references

Core readings	Suggested readings
<ul style="list-style-type: none">• [Dennis15] – Chap. 3	<ul style="list-style-type: none">• [Larman'12]– Chap. 5• [Pressman'10] – Chap. 5