UNIFIED MODELING LANGUAGE

Adrián A. Alarcón O.

UML

Lenguaje de Modelado Unificado

Lenguaje estandar para especificar, visualizar, construir y documentar elementos de sistemas de software

UML

- Creado por Object Management Group OMG
- UML 1.0 Enero 1997
- Lenguaje Pictórico
- Proposito General

CONCEPTOS ORIENTADOS A OBJETOS

- Clases
- Objectos
- Abstracción
- Encapsulamento
- Herencia
- Polimorfismo

DIAGRAMA DE CLASES

DIAGRAMA DE CLASES

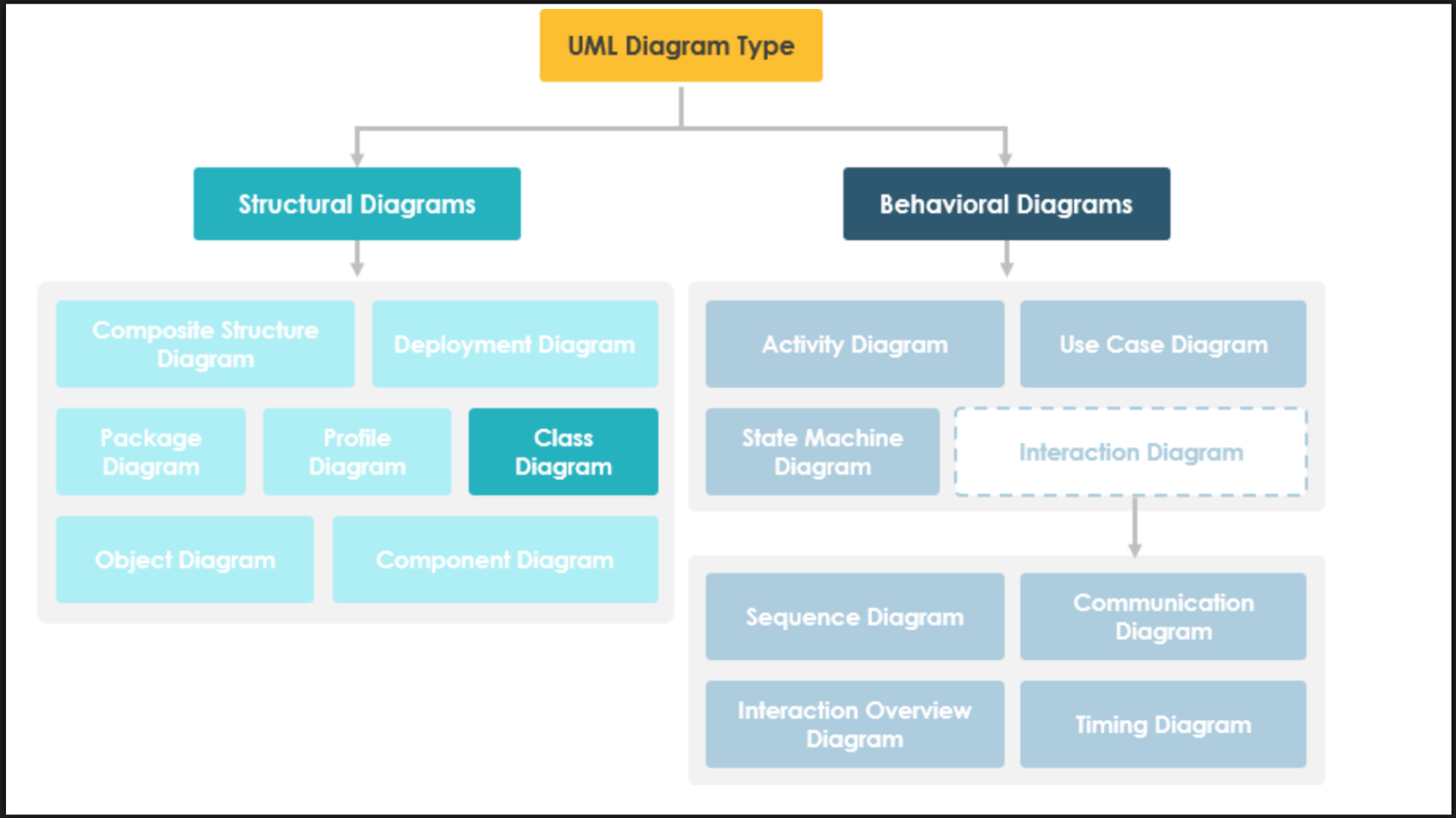
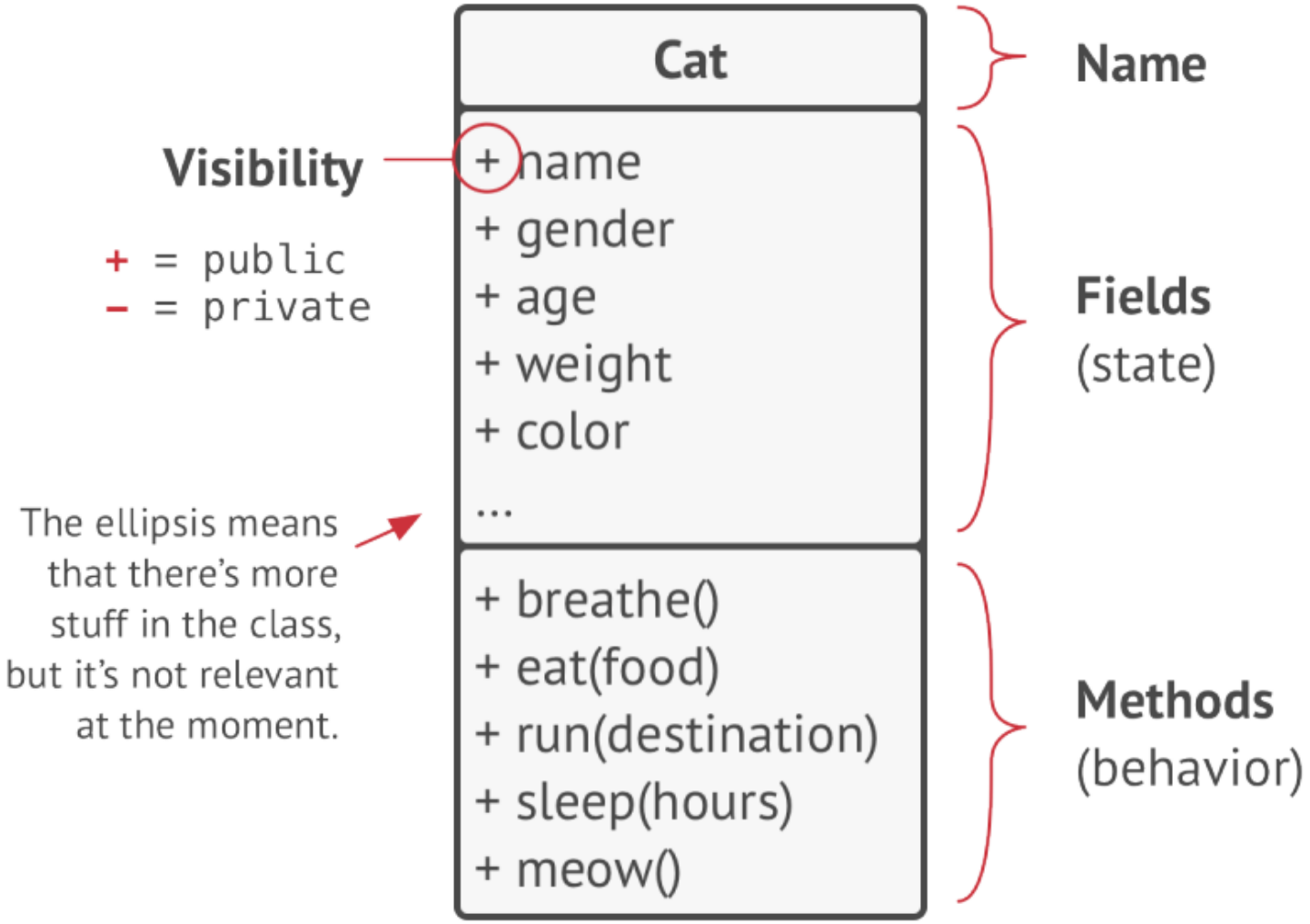


DIAGRAMA DE CLASES

Representa una vista estatica de una aplicación desde el punto de vista de las clases del sistem

- Atributos
- Operaciones
- Relaciones

CLASE


Cat
Name
Visibility
+ = public
– = private
+ name
+ gender
+ age
+ weight
+ color
...
Fields
(state)
The ellipsis means that there's more stuff in the class, but it's not relevant at the moment.
+ breathe()
+ eat(food)
+ run(destination)
+ sleep(hours)
+ meow()
Methods
(behavior)

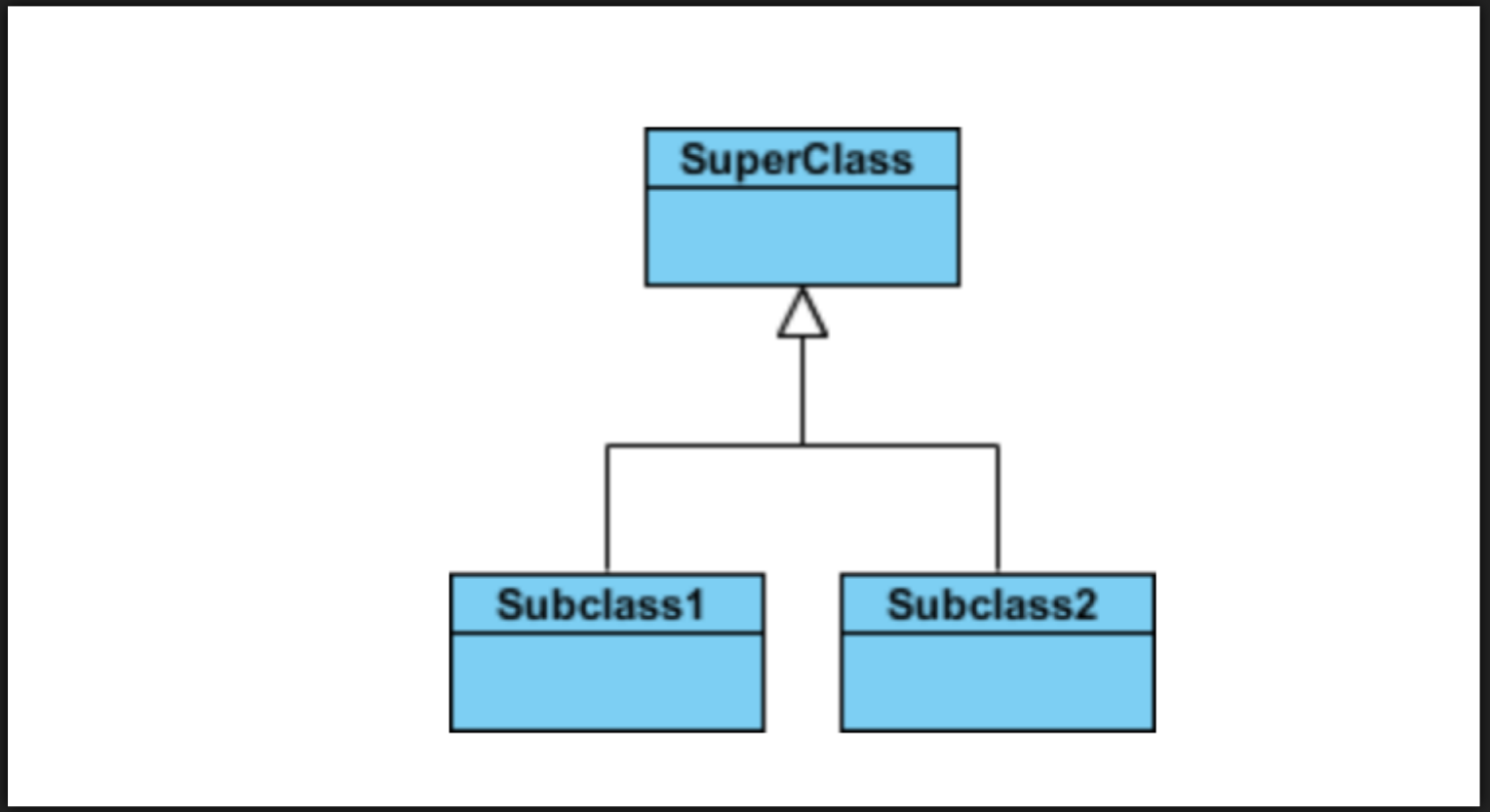
RELACIONES

- Herencia
- Asociación Simple
- Agregación
- Composición
- Dependencia

HERENCIA

- Flecha en Blanco
- "Es Un"
- Clases Abstractas en Italicas
- Interfaces con Etiqueta "Interface"

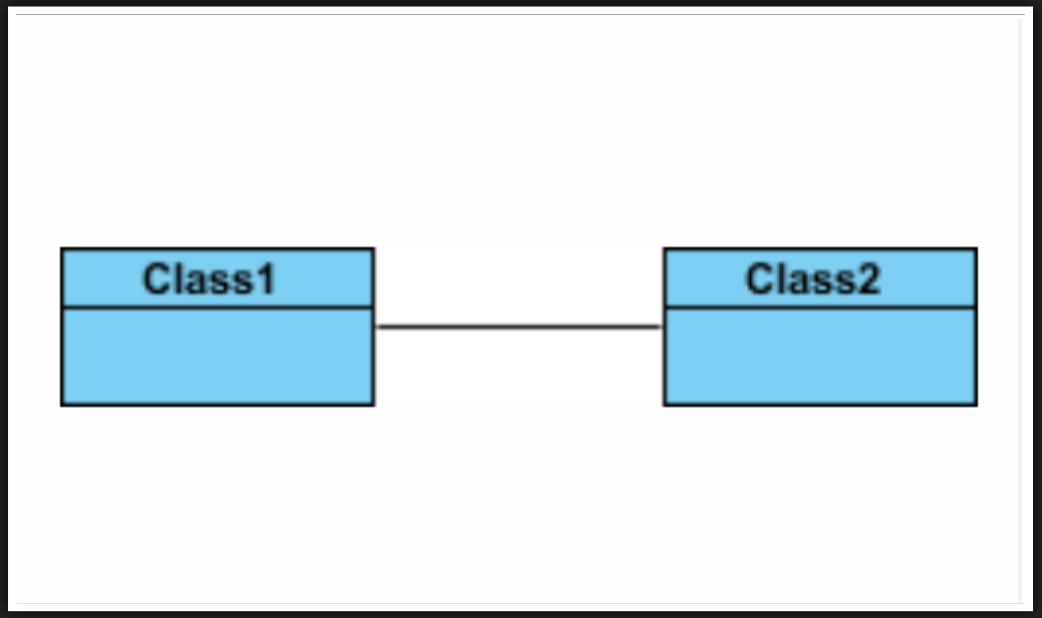
HERENCIA



ASOCIACIÓN SIMPLE

- Línea Solida
- Enlace Entre Dos Clases

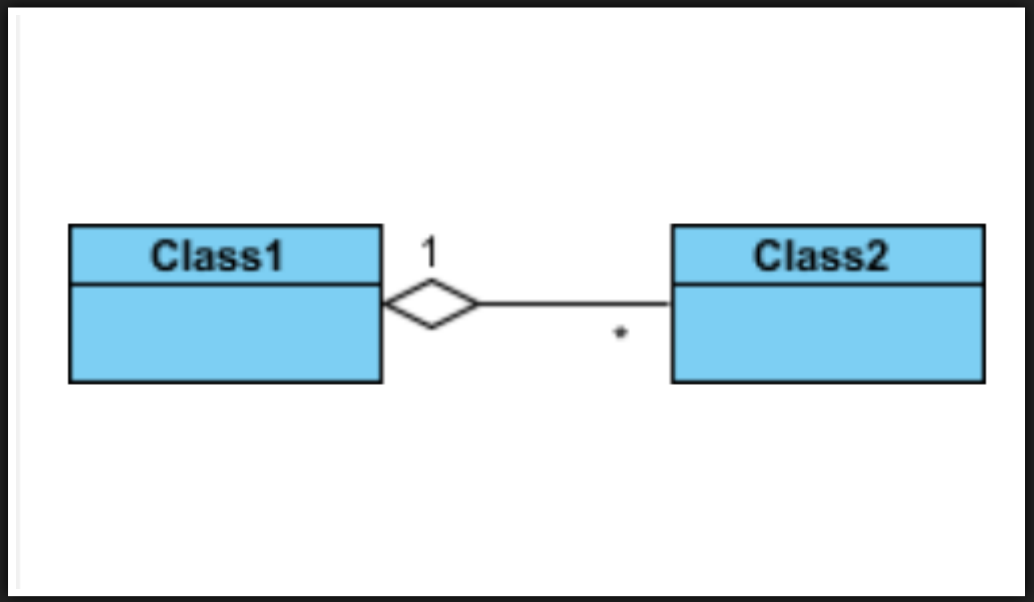
ASOCIACIÓN SIMPLE



AGREGACIÓN

- Línea Solida con Punta de Rombo Blanco
- "Es Parte De" o "Consiste De"

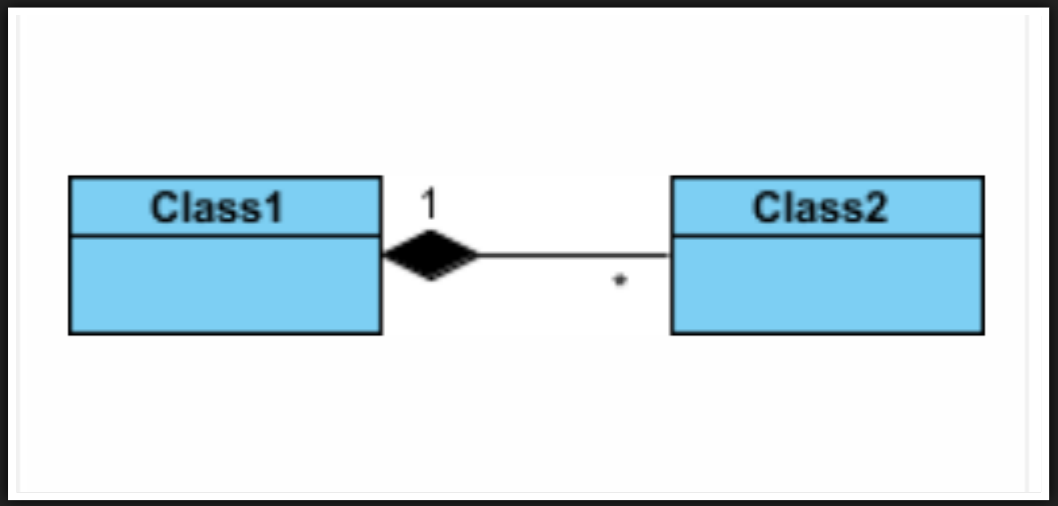
AGREGACIÓN



COMPOSICIÓN

- Línea Solida con Punta de Rombo Negro
- "Se Compone De"
- No Puede Existir Por Si Solo

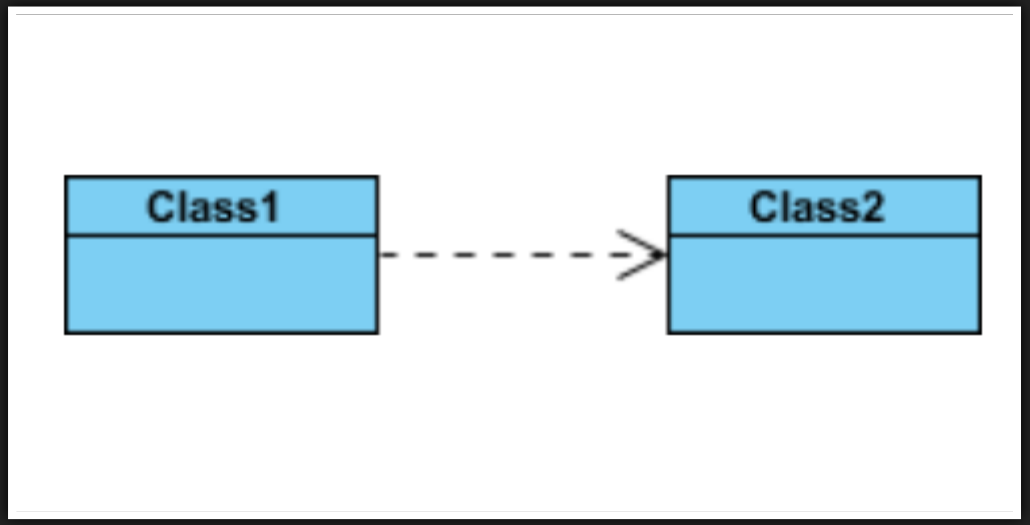
COMPOSICIÓN



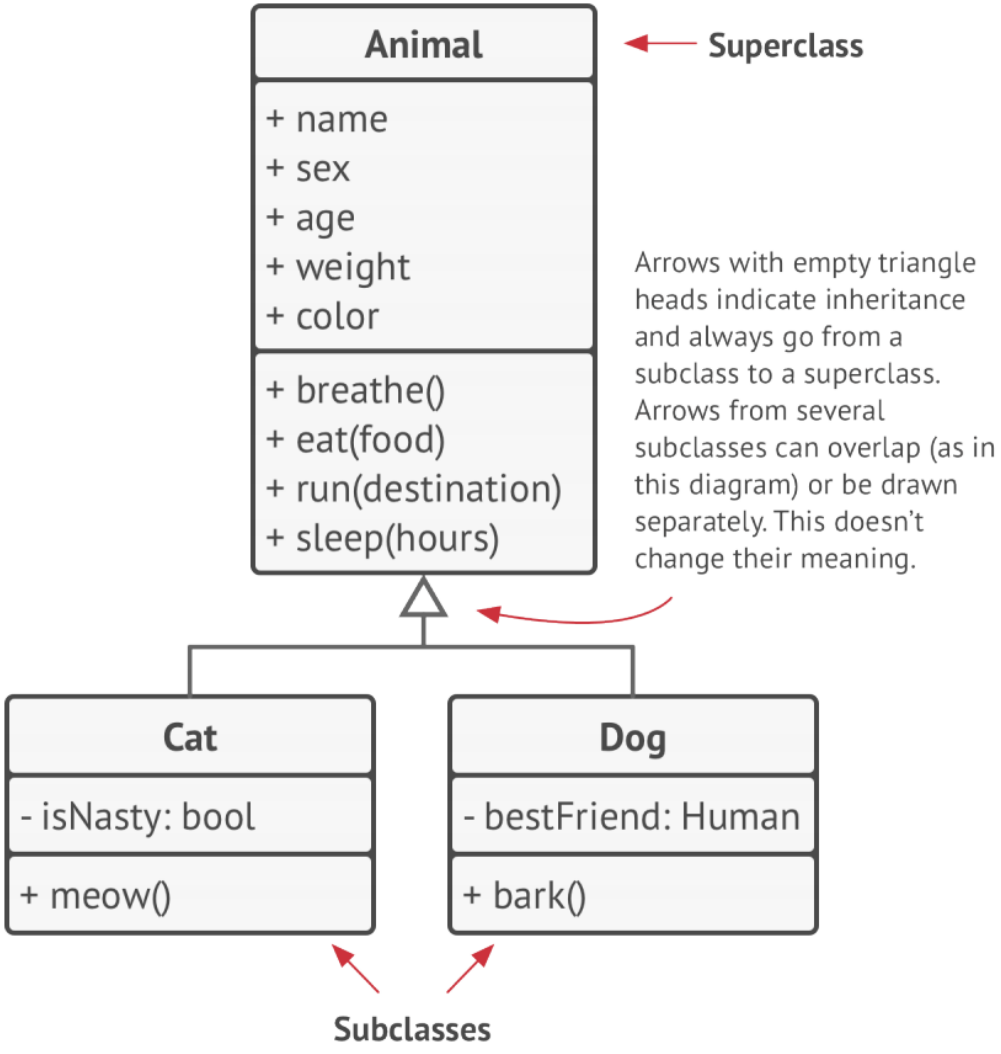
DEPENDENCIA

- Línea Pnteada con Flecha
- "Depende De"
- Dependencia Entre Clases

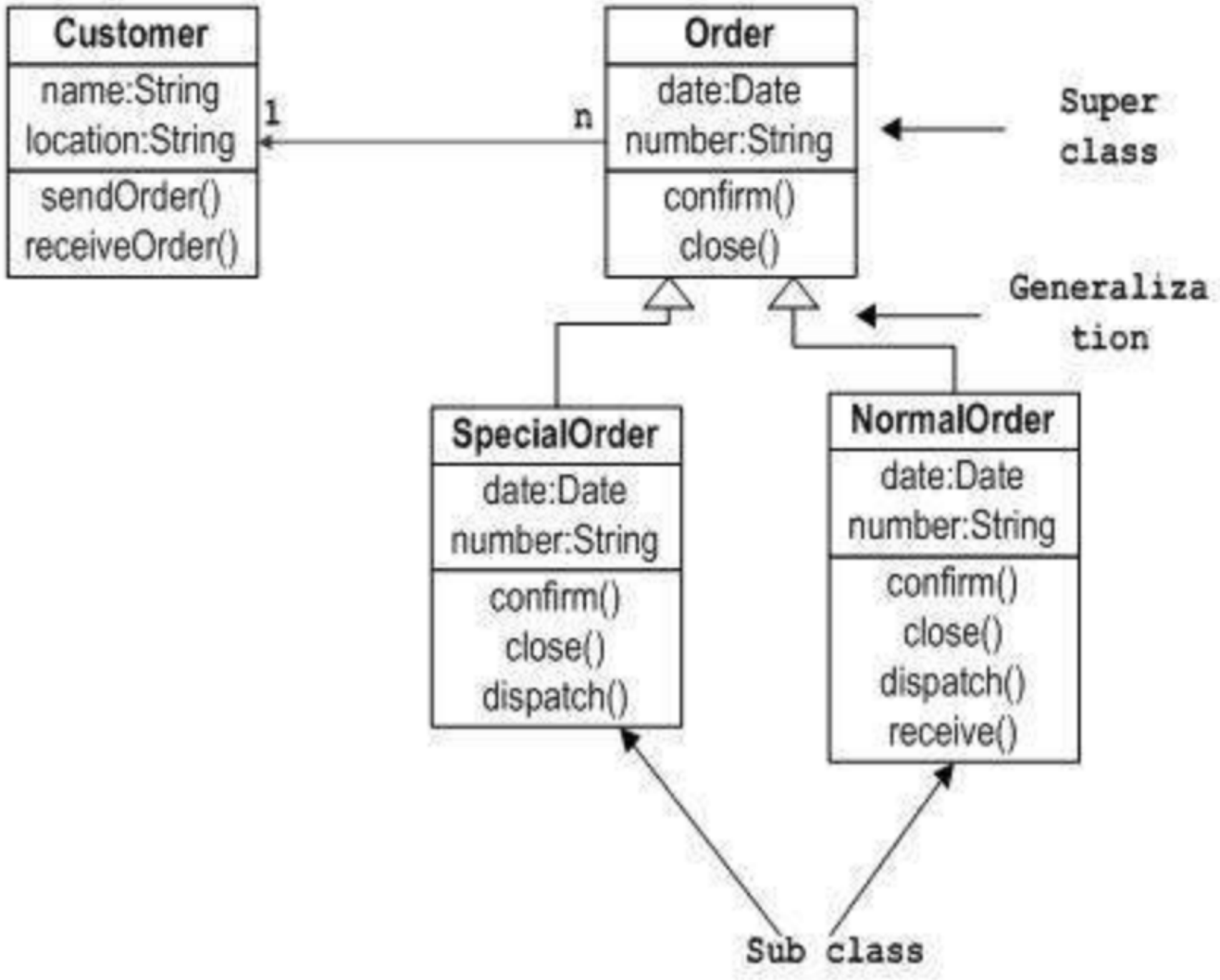
DEPENDENCIA



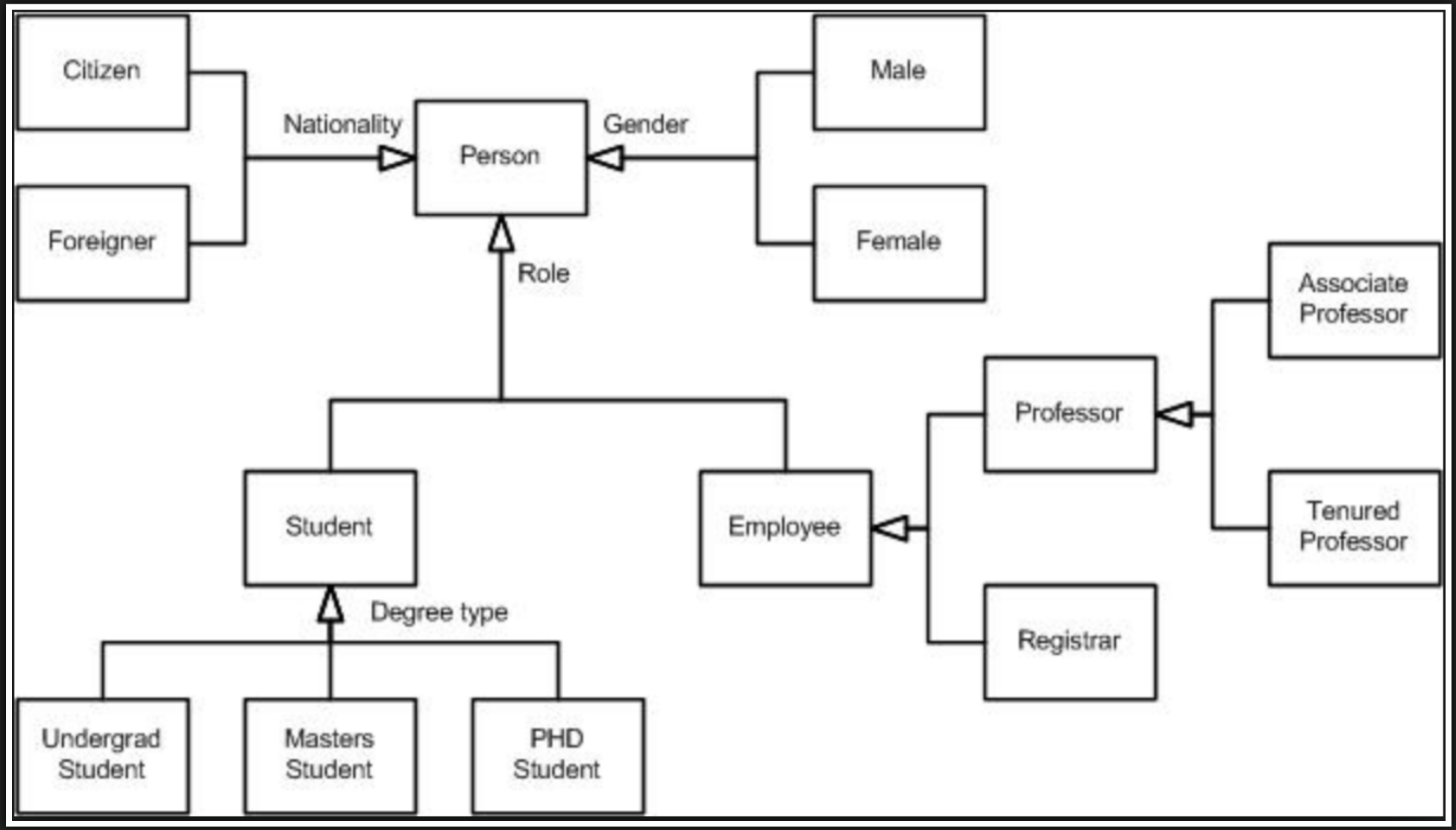
EJEMPLOS


Animal
+ name
+ sex
+ age
+ weight
+ color
+ breathe()
+ eat(food)
+ run(destination)
+ sleep(hours)
Superclass
Arrows with empty triangle heads indicate inheritance and always go from a subclass to a superclass. Arrows from several subclasses can overlap (as in this diagram) or be drawn separately. This doesn't change their meaning.
Cat
- isNasty: bool
+ meow()
Dog
- bestFriend: Human
+ bark()
Subclasses

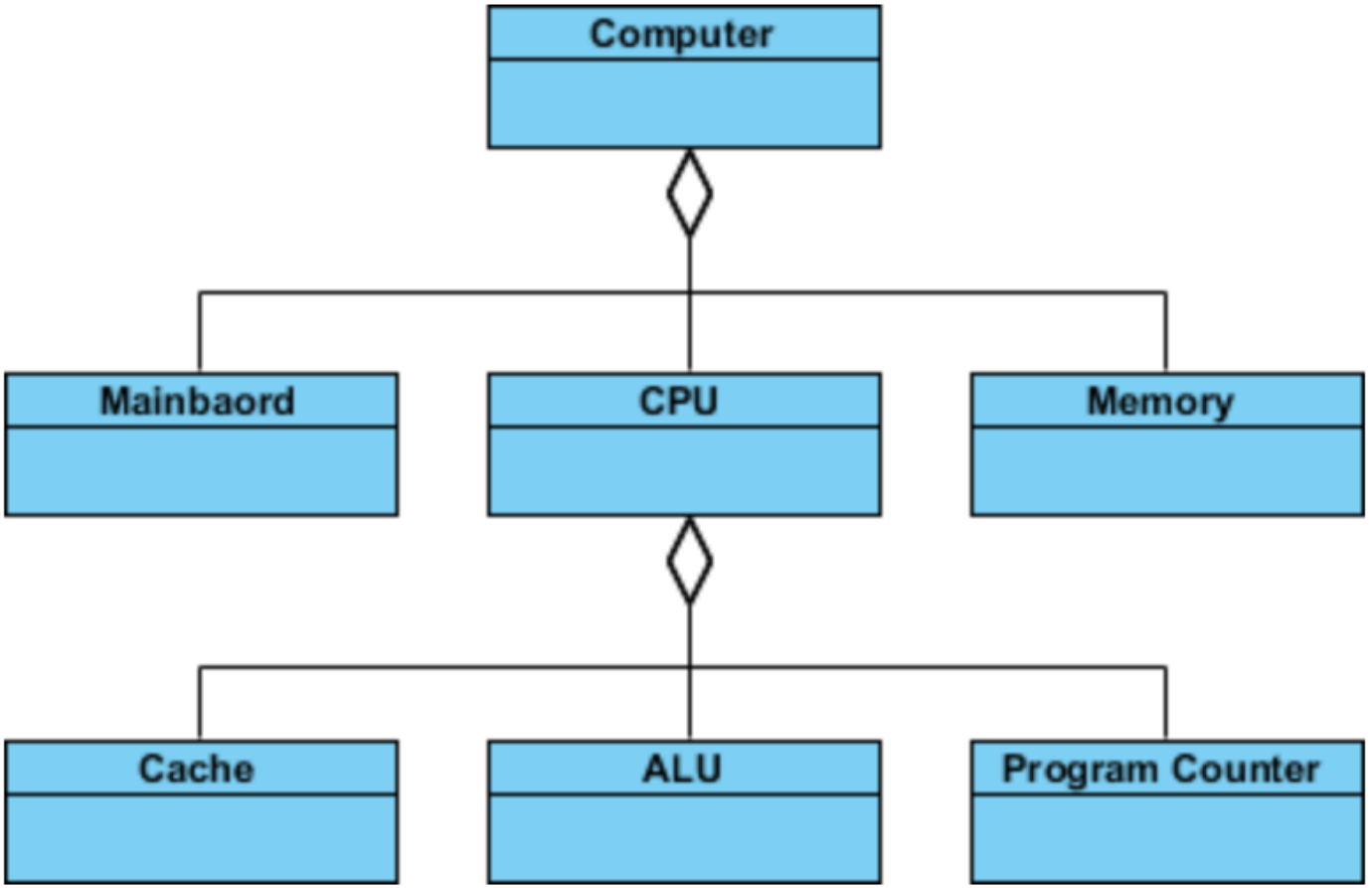
EJEMPLOS


Customer
name:String
location:String
sendOrder()
receiveOrder()
1
n
Order
date:Date
number:String
confirm()
close()
Super class
Generalization
SpecialOrder
date:Date
number:String
confirm()
close()
dispatch()
NormalOrder
date:Date
number:String
confirm()
close()
dispatch()
receive()
Sub class

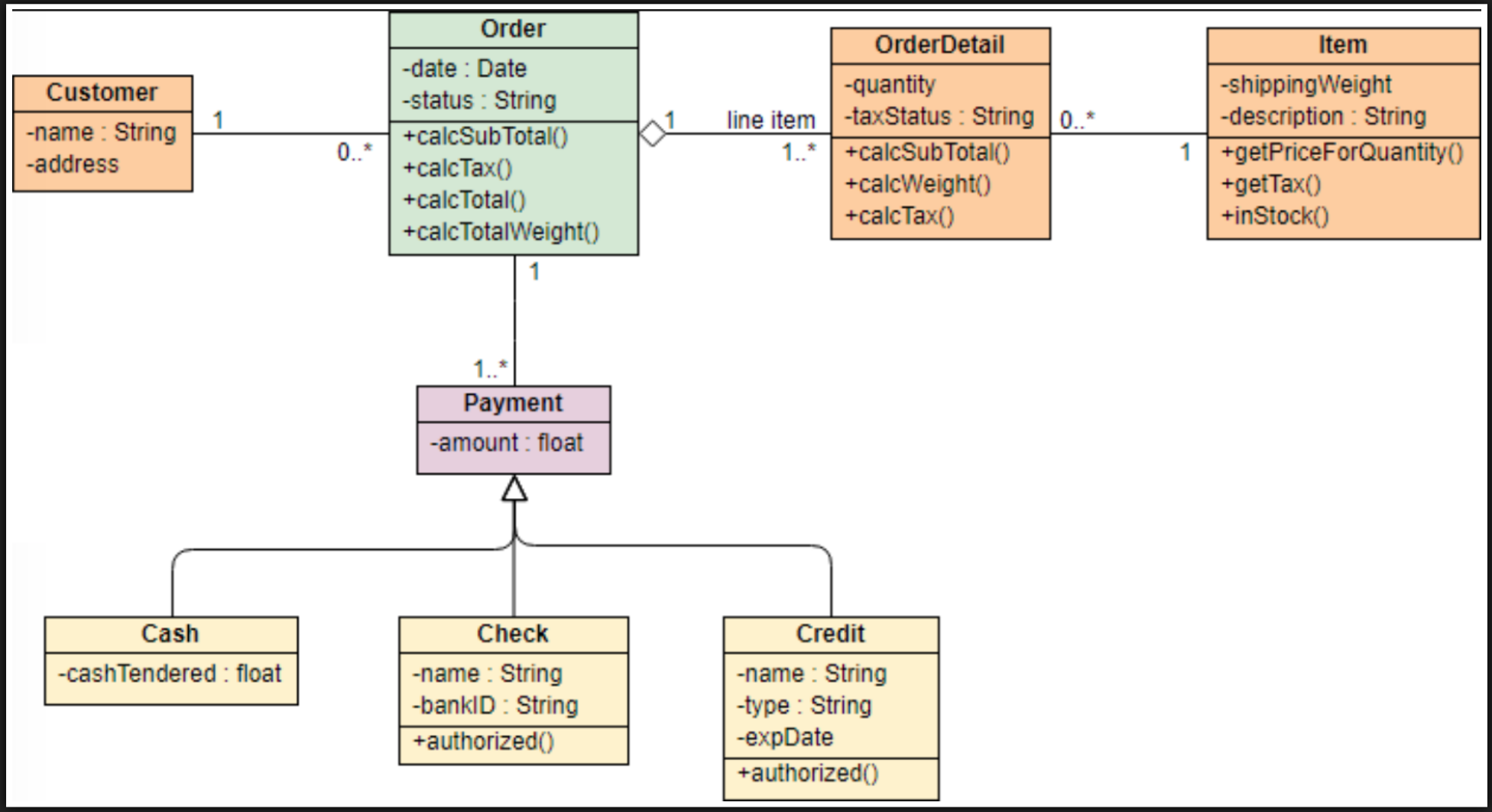
EJEMPLOS


Citizen
Foreigner
Nationality
Person
Gender
Male
Female
Role
Student
Degree type
Undergrad Student
Masters Student
PHD Student
Employee
Professor
Associate Professor
Tenured Professor
Registrar

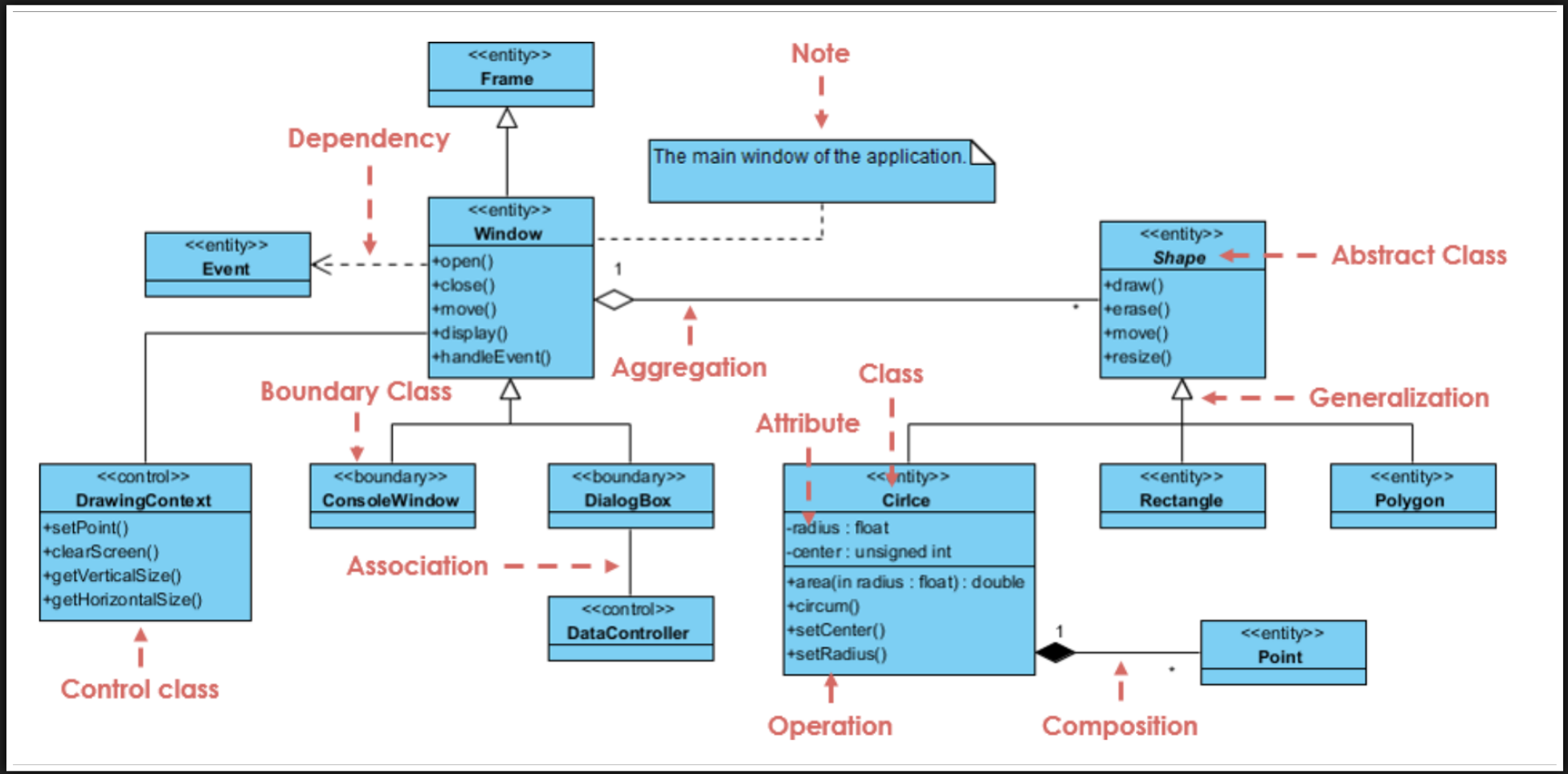
EJEMPLOS



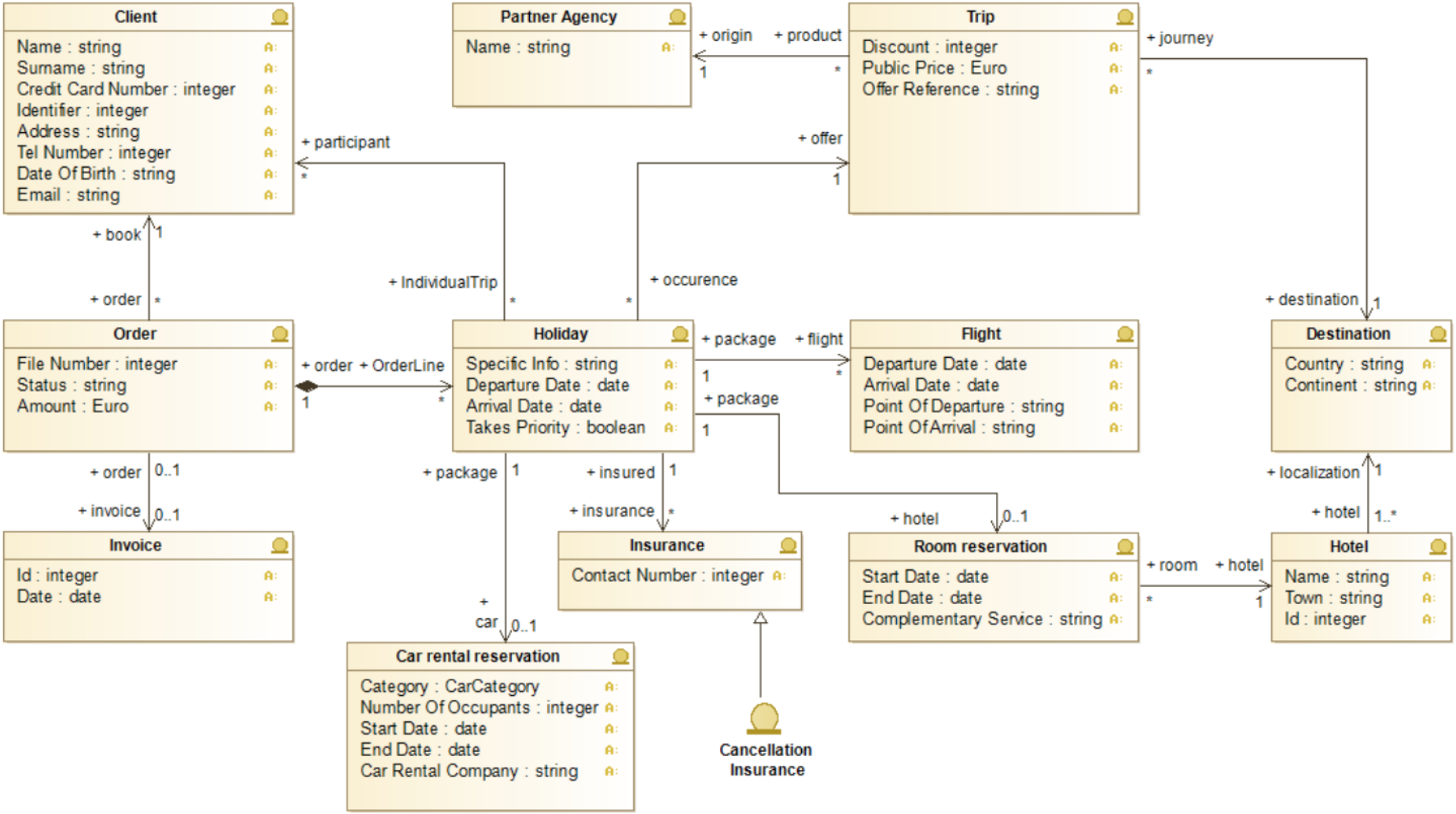
EJEMPLOS


Customer
-name : String
-address
1
0..*
Order
-date : Date
-status : String
+calcSubTotal()
+calcTax()
+calcTotal()
+calcTotalWeight()
1
line item
1..*
OrderDetail
-quantity
-taxStatus : String
+calcSubTotal()
+calcWeight()
+calcTax()
0..*
1
Item
-shippingWeight
-description : String
+getPriceForQuantity()
+getTax()
+inStock()
1
1..*
Payment
-amount : float
Cash
-cashTendered : float
Check
-name : String
-bankID : String
+authorized()
Credit
-name : String
-type : String
-expDate
+authorized()

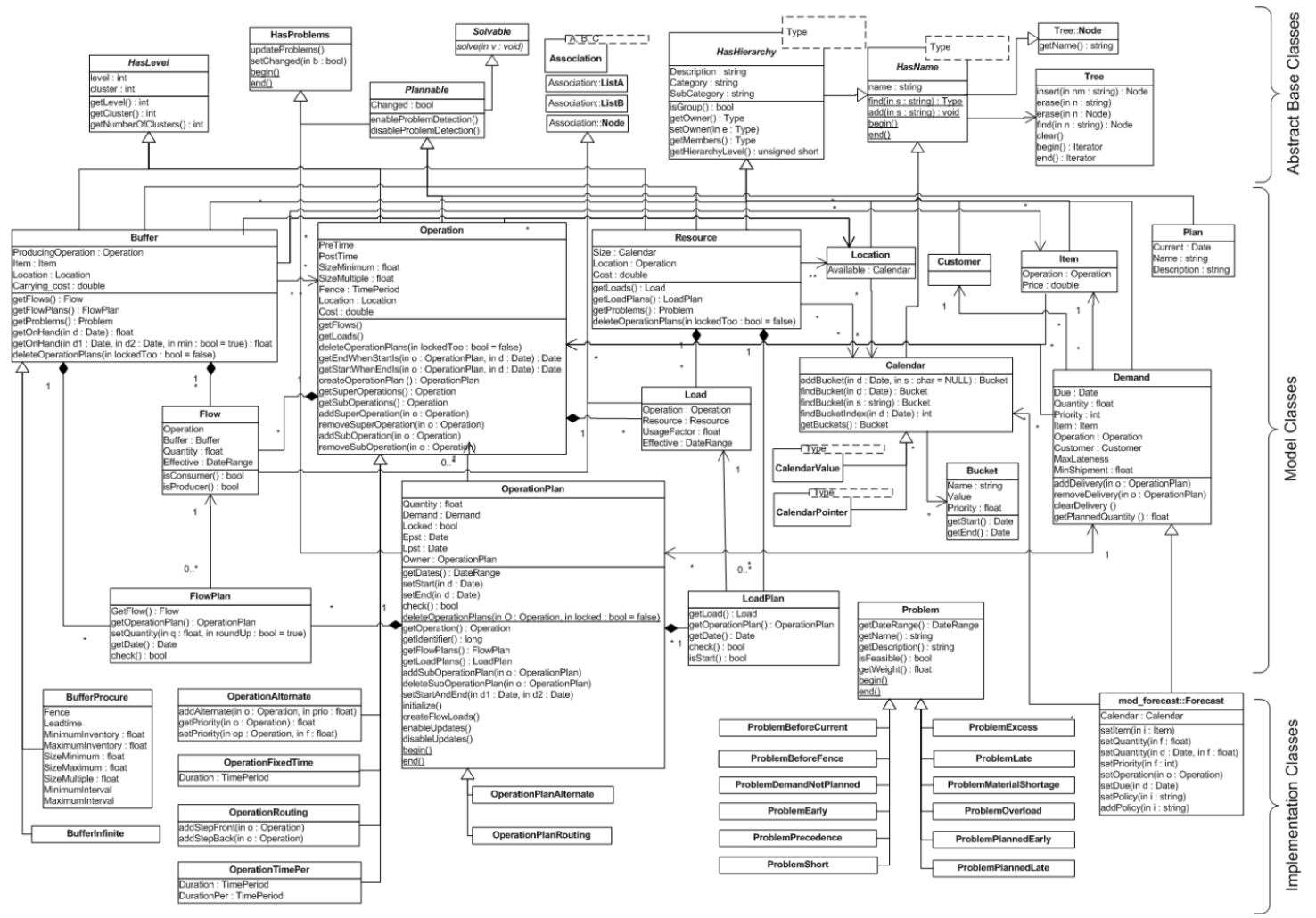
EJEMPLOS


<<entity>>
Frame
Note
Dependency
The main window of the application.
<<entity>>
Window
+open()
+close()
+move()
+display()
+handleEvent()
<<entity>>
Event
1
<<entity>>
Shape
+draw()
+erase()
+move()
+resize()
Abstract Class
Aggregation
Boundary Class
Class
Attribute
Generalization
<<control>>
DrawingContext
+setPoint()
+clearScreen()
+getVerticalSize()
+getHorizontalSize()
<<boundary>>
ConsoleWindow
<<boundary>>
DialogBox
<<entity>>
Circle
-radius : float
-center : unsigned int
+area(in radius : float) : double
+circum()
+setCenter()
+setRadius()
<<entity>>
Rectangle
<<entity>>
Polygon
Association
<<control>>
DataController
1
<<entity>>
Point
Control class
Operation
Composition

EJEMPLOS


Client
Name : string
Surname : string
Credit Card Number : integer
Identifier : integer
Address : string
Tel Number : integer
Date Of Birth : string
Email : string
Partner Agency
Name : string
Trip
Discount : integer
Public Price : Euro
Offer Reference : string
+ origin
+ product
+ journey
+ participant
+ offer
+ book
+ order
+ IndividualTrip
+ occurence
+ destination
Order
File Number : integer
Status : string
Amount : Euro
+ order
+ OrderLine
Holiday
Specific Info : string
Departure Date : date
Arrival Date : date
Takes Priority : boolean
+ package
+ flight
Flight
Departure Date : date
Arrival Date : date
Point Of Departure : string
Point Of Arrival : string
Destination
Country : string
Continent : string
+ order
+ invoice
+ package
+ insured
+ insurance
+ localization
+ hotel
Invoice
Id : integer
Date : date
Insurance
Contact Number : integer
Room reservation
Start Date : date
End Date : date
Complementary Service : string
+ room
+ hotel
Hotel
Name : string
Town : string
Id : integer
+ car
Car rental reservation
Category : CarCategory
Number Of Occupants : integer
Start Date : date
End Date : date
Car Rental Company : string
Cancellation Insurance

EJEMPLOS


Abstract Base Classes
Model Classes
Implementation Classes
HasProblems
updateProblems()
setChanged(in b : bool)
begin()
end()
Solvable
solve(in v : void)
Association
Association::ListA
Association::ListB
Association::Node
HasHierarchy
Description : string
Category : string
SubCategory : string
isGroup() : bool
getOwner() : Type
setOwner(in e : Type)
getMembers() : Type
getHierarchyLevel() : unsigned short
HasName
name : string
find(in s : string) : Type
add(in s : string) : void
begin()
end()
Tree::Node
getName() : string
Tree
insert(in nm : string) : Node
erase(in n : string)
erase(in n : Node)
find(in n : string) : Node
clear()
begin() : Iterator
end() : Iterator
HasLevel
level : int
cluster : int
getLevel() : int
getCluster() : int
getNumberOfClusters() : int
Plannable
Changed : bool
enableProblemDetection()
disableProblemDetection()
Buffer
ProducingOperation : Operation
Item : Item
Location : Location
Carrying_cost : double
getFlows() : Flow
getFlowPlans() : FlowPlan
getProblems() : Problem
getOnHand(in d : Date) : float
getOnHand(in d1 : Date, in d2 : Date, in min : bool = true) : float
deleteOperationPlans(in lockedToo : bool = false)
Operation
PreTime
PostTime
SizeMinimum : float
SizeMultiple : float
Fence : TimePeriod
Location : Location
Cost : double
getFlows()
getLoads()
deleteOperationPlans(in lockedToo : bool = false)
getEndWhenStartIs(in o : OperationPlan, in d : Date) : Date
getStartWhenEndIs(in o : OperationPlan, in d : Date) : Date
createOperationPlan () : OperationPlan
getSuperOperations() : Operation
getSubOperations() : Operation
addSuperOperation(in o : Operation)
removeSuperOperation(in o : Operation)
addSubOperation(in o : Operation)
removeSubOperation(in o : Operation)
Resource
Size : Calendar
Location : Operation
Cost : double
getLoads() : Load
getLoadPlans() : LoadPlan
getProblems() : Problem
deleteOperationPlans(in lockedToo : bool = false)
Location
Available : Calendar
Customer
Item
Operation : Operation
Price : double
Plan
Current : Date
Name : string
Description : string
Flow
Operation
Buffer : Buffer
Quantity : float
Effective : DateRange
isConsumer() : bool
isProducer() : bool
Load
Operation : Operation
Resource : Resource
UsageFactor : float
Effective : DateRange
Calendar
addBucket(in d : Date, in s : char = NULL) : Bucket
findBucket(in d : Date) : Bucket
findBucket(in s : string) : Bucket
findBucketIndex(in d : Date) : int
getBuckets() : Bucket
CalendarValue
CalendarPointer
Bucket
Name : string
Value
Priority : float
getStart() : Date
getEnd() : Date
Demand
Due : Date
Quantity : float
Priority : int
Item : Item
Operation : Operation
Customer : Customer
MaxLateness
MinShipment : float
addDelivery(in o : OperationPlan)
removeDelivery(in o : OperationPlan)
clearDelivery ()
getPlannedQuantity () : float
OperationPlan
Quantity : float
Demand : Demand
Locked : bool
Epst : Date
Lpst : Date
Owner : OperationPlan
getDates() : DateRange
setStart(in d : Date)
setEnd(in d : Date)
check() : bool
deleteOperationPlans(in O : Operation, in locked : bool = false)
getOperation() : Operation
getIdentifier() : long
getFlowPlans() : FlowPlan
getLoadPlans() : LoadPlan
addSubOperationPlan(in o : OperationPlan)
deleteSubOperationPlan(in o : OperationPlan)
setStartAndEnd(in d1 : Date, in d2 : Date)
initialize()
createFlowLoads()
enableUpdates()
disableUpdates()
begin()
end()
FlowPlan
GetFlow() : Flow
getOperationPlan() : OperationPlan
setQuantity(in q : float, in roundUp : bool = true)
getDate() : Date
check() : bool
LoadPlan
getLoad() : Load
getOperationPlan() : OperationPlan
getDate() : Date
check() : bool
isStart() : bool
Problem
getDateRange() : DateRange
getName() : string
getDescription() : string
isFeasible() : bool
getWeight() : float
begin()
end()
mod_forecast::Forecast
Calendar : Calendar
setItem(in i : Item)
setQuantity(in f : float)
setQuantity(in d : Date, in f : float)
setPriority(in f : int)
setOperation(in o : Operation)
setDue(in d : Date)
setPolicy(in i : string)
addPolicy(in i : string)
BufferProcure
Fence
Leadtime
MinimumInventory : float
MaximumInventory : float
SizeMinimum : float
SizeMaximum : float
SizeMultiple : float
MinimumInterval
MaximumInterval
BufferInfinite
OperationAlternate
addAlternate(in o : Operation, in prio : float)
getPriority(in o : Operation) : float
setPriority(in op : Operation, in f : float)
OperationFixedTime
Duration : TimePeriod
OperationRouting
addStepFront(in o : Operation)
addStepBack(in o : Operation)
OperationTimePer
Duration : TimePeriod
DurationPer : TimePeriod
OperationPlanAlternate
OperationPlanRouting
ProblemBeforeCurrent
ProblemBeforeFence
ProblemDemandNotPlanned
ProblemEarly
ProblemPrecedence
ProblemShort
ProblemExcess
ProblemLate
ProblemMaterialShortage
ProblemOverload
ProblemPlannedEarly
ProblemPlannedLate

EJERCICIO

https://bit.ly/2RRMKMJ

https://www.draw.io/