UML

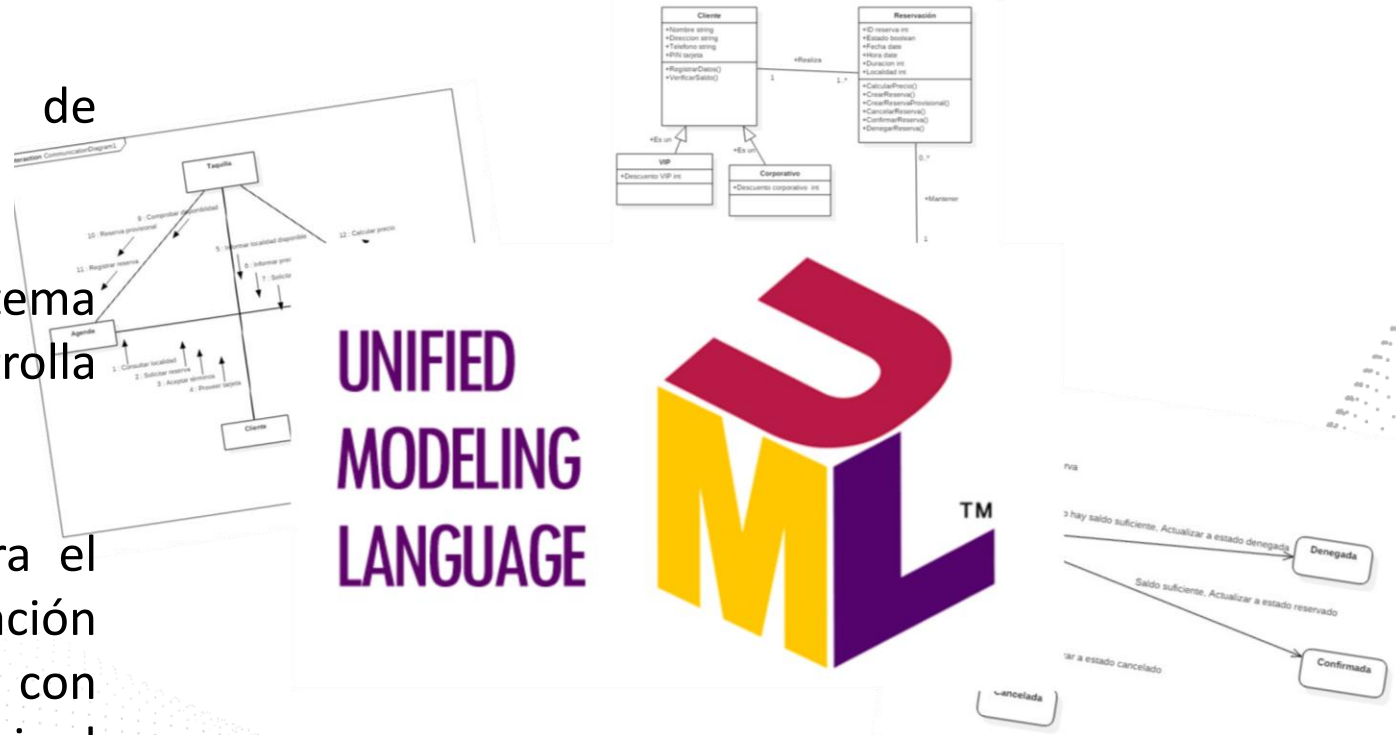
Lenguaje de Modelado Unificado

UML – Lenguaje de Modelado Unificado

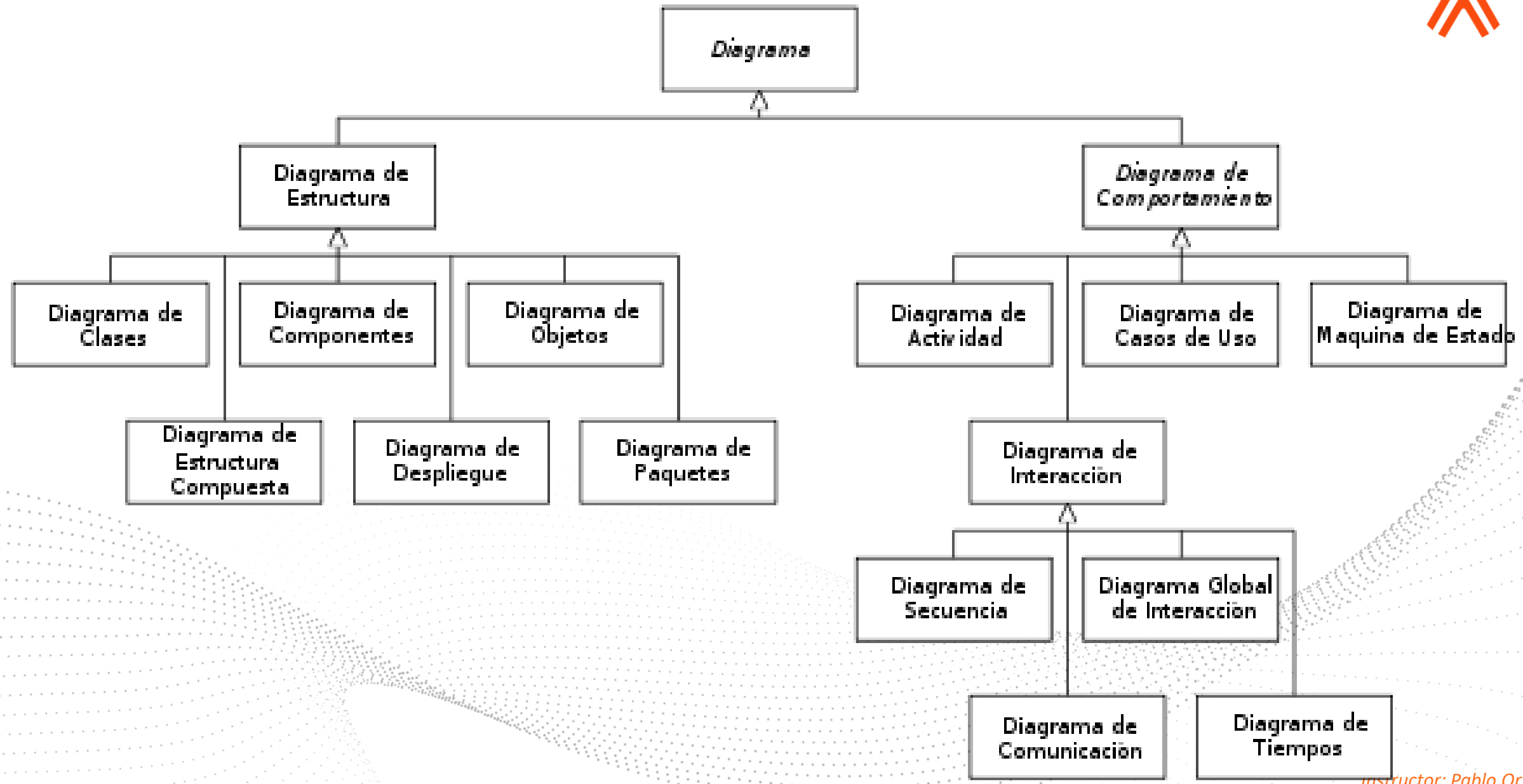
UML (Unified Model Language) o Lenguaje de Modelamiento Unificado.

Este paradigma permite diagramar todo un sistema de información al igual que un arquitecto desarrolla sus planos para luego llevarlos al terreno.

UML es un lenguaje de notación estándar para el análisis y modelado de un sistema de información orientado a objetos (POO). No debe confundirse con un método, debido a que no indica cómo construir el sistema.



UML – Tipos de Diagramas



Diagramas de Clase

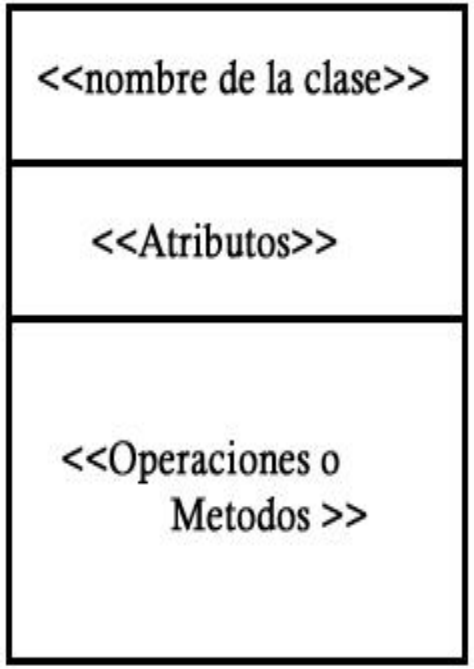
Instructor: Pablo Ortiz

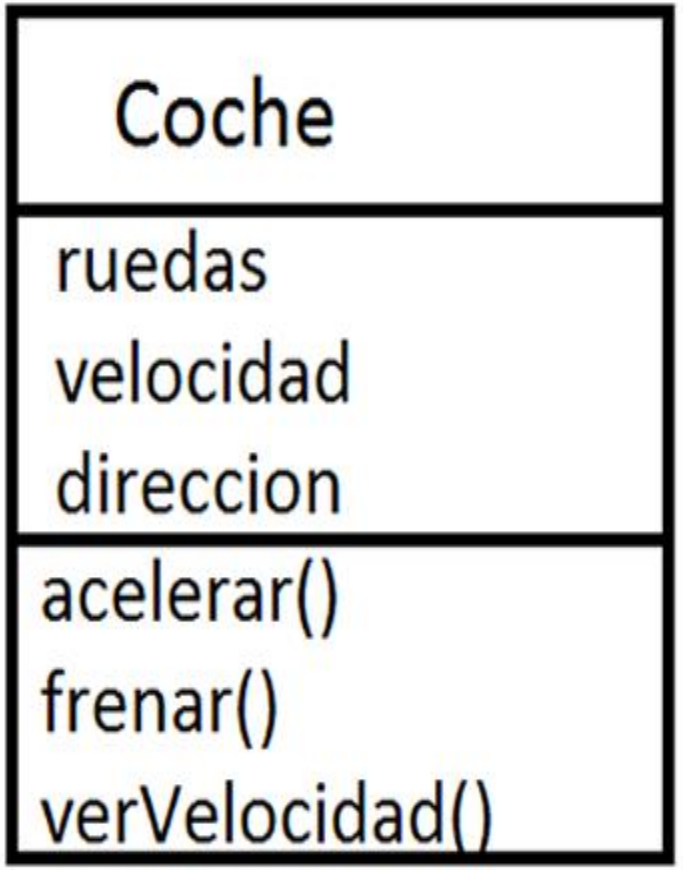
CONCEPTOS

Clase: Es la descripción de un conjunto de objetos que comparten atributos, método, relaciones, etc.

Atributo: Son los valores que corresponden a u objeto como olor, material, cantidad, ubicación, etc. Y se conoce como la información detallada del objeto.

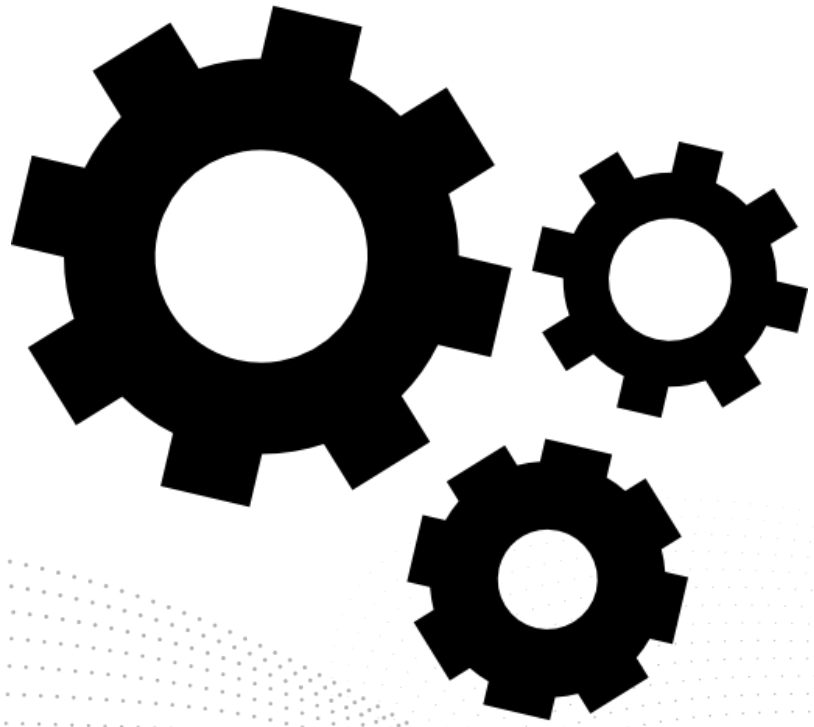
Métodos: (operación), comportamiento asociado a una clase que corresponde a las funciones que cumple un objeto



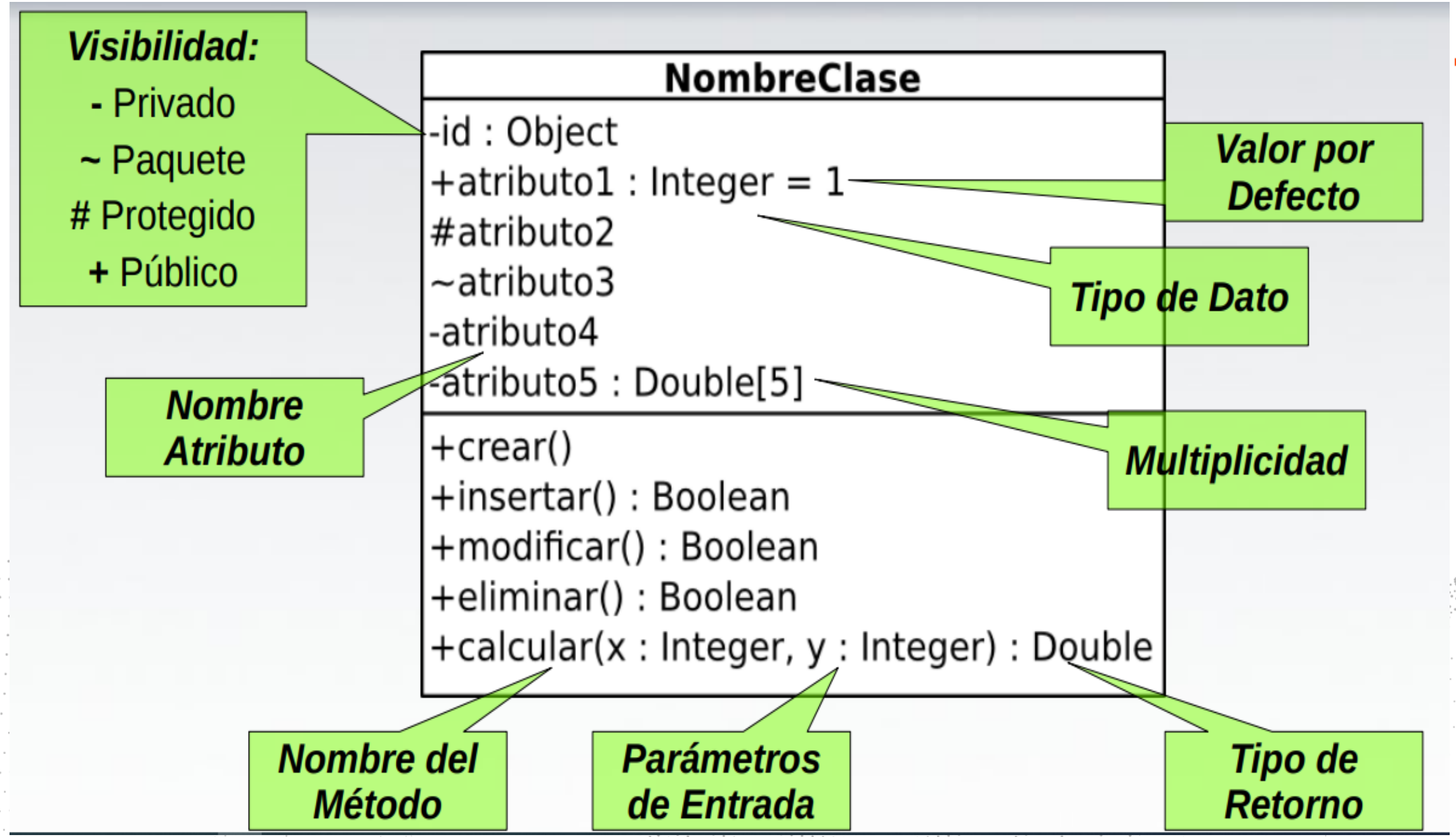


CLIENTE	clase
Edad Nombre Documento Genero Email Teléfono	Atributos
Comprar Vender Registro	Métodos

FUNCION



- Describir la estructura del sistema donde se mostraran sus clases, atributos y las relaciones que existen entre estos.
- Visualizar las relaciones que existen entre las clases involucradas en el sistema.
- Modelar el esquema lógico de una Base de Datos
- Estos diagramas son utilizados en la etapa de análisis y diseño, en primera instancia se plantea el diseño conceptual de la información que se utiliza dentro del sistema y luego se desarrolla la parte de componentes que se encargan de las relaciones entre unos y otros.


Visibilidad:
- Privado
~ Paquete
# Protegido
+ Público
NombreClase
-id : Object
+atributo1 : Integer = 1
#atributo2
~atributo3
-atributo4
-atributo5 : Double[5]
+crear()
+insertar() : Boolean
+modificar() : Boolean
+eliminar() : Boolean
+calcular(x : Integer, y : Integer) : Double
Valor por Defecto
Tipo de Dato
Nombre Atributo
Multiplicidad
Nombre del Método
Parámetros de Entrada
Tipo de Retorno

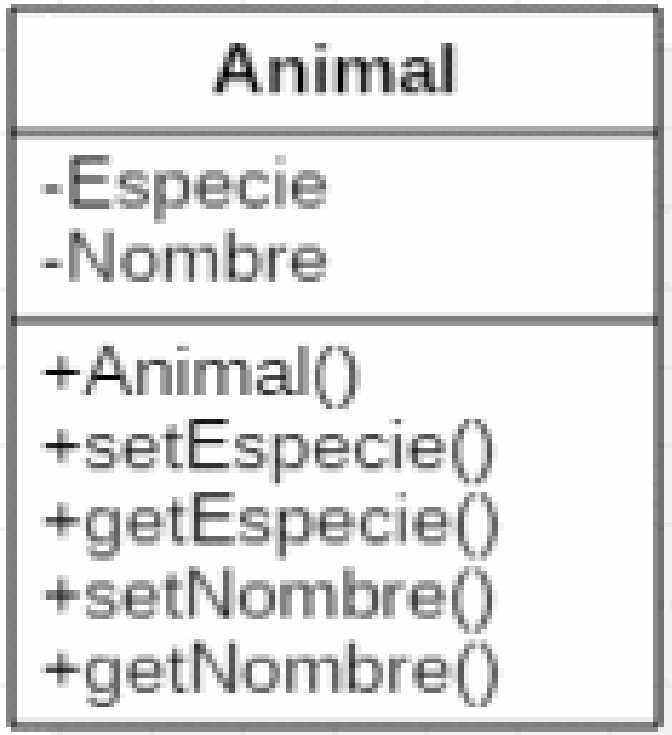
ATRIBUTOS

Son valores o atributos que corresponden una clase

Tanto los atributos como las funciones incluyen al principio de su descripción la visibilidad que tendrá. Esta visibilidad se identifica escribiendo un símbolo y podrá ser:

- **(+) Pública.** Representa que se puede acceder al atributo o función desde cualquier lugar de la aplicación.
- **(-) Privada.** Representa que se puede acceder al atributo o función únicamente desde la misma clase.
- **(#) Protegida.** Representa que el atributo o función puede ser accedida únicamente desde la misma clase o desde las clases que hereden de ella (clases derivadas).

Estos tres tipos de visibilidad son los más comunes. No obstante, pueden incluirse otros en base al lenguaje de programación que se esté usando (no es muy común). Por ejemplo: (/) Derivado o (~) Paquete.

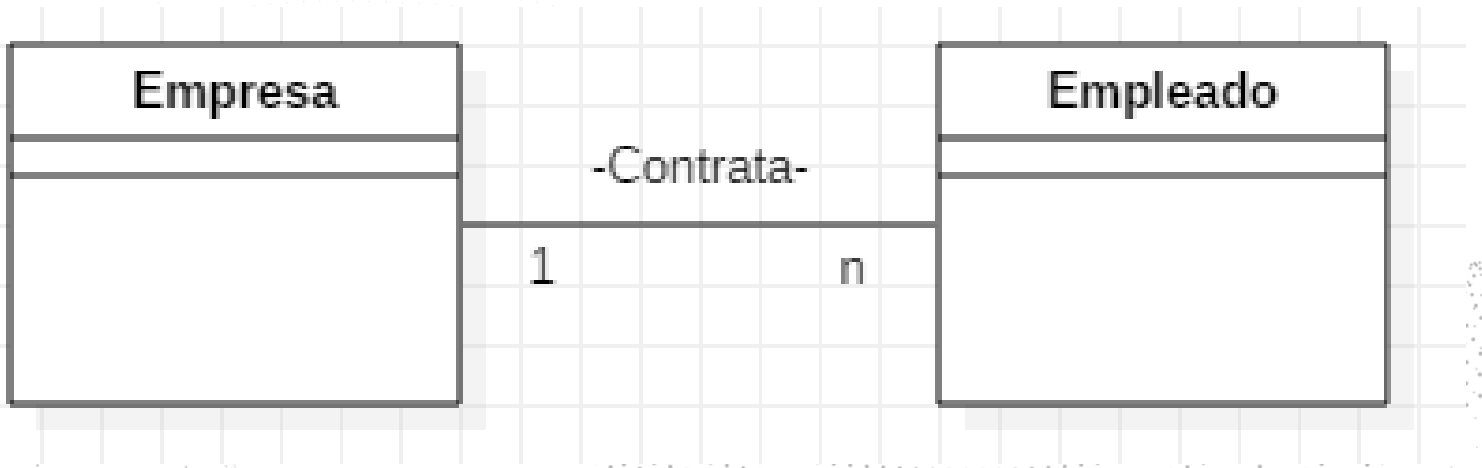


RELACION

Una relación i**dentifica una dependencia**. Esta dependencia puede ser entre dos o más clases. Las relaciones se representan con una línea que une las clases, esta línea variará dependiendo del tipo de relación.

Las relaciones en el diagrama de clases tienen varias propiedades, que dependiendo la profundidad que se quiera dar al diagrama se representarán o no. Estas propiedades son las siguientes:

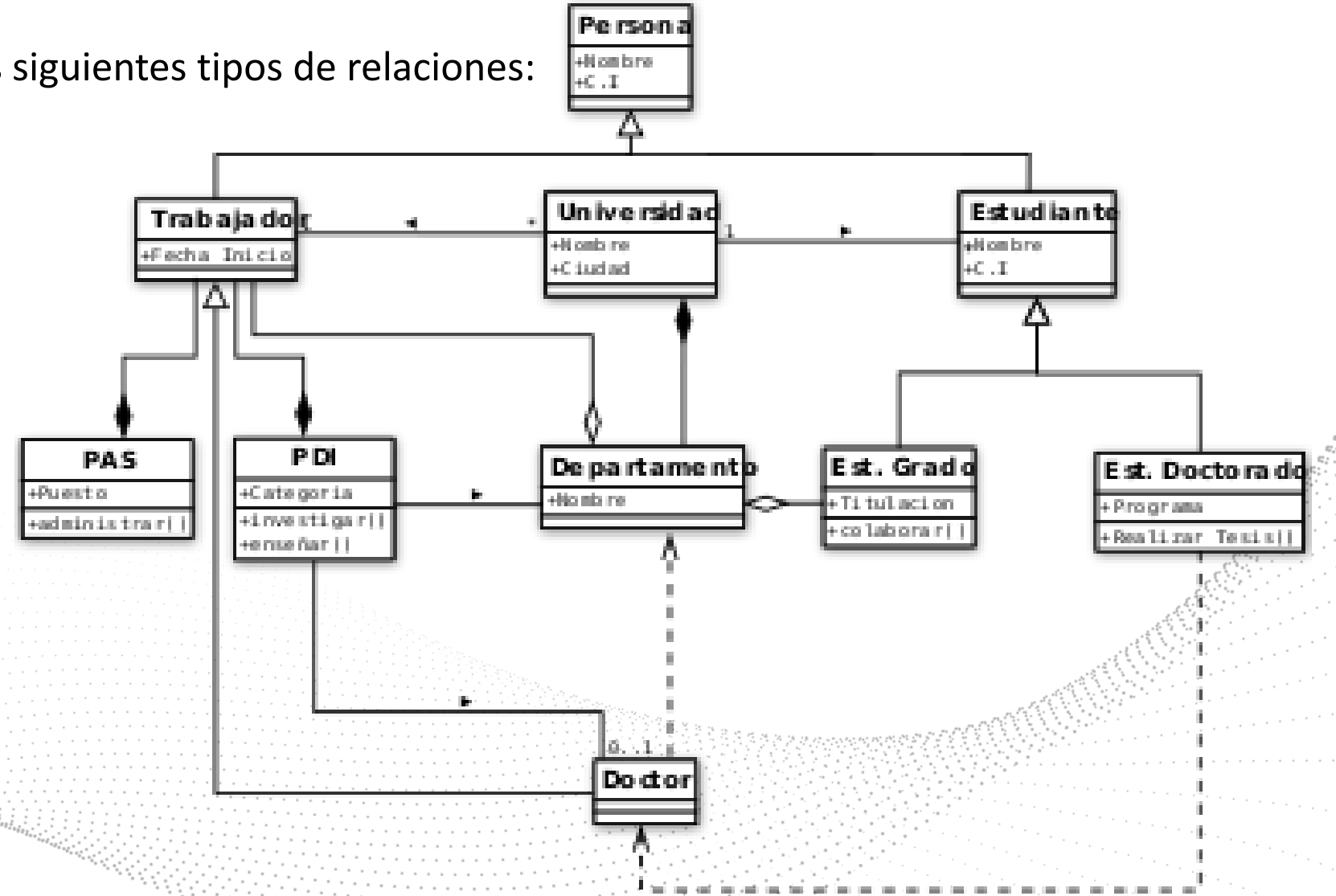
- **Multiplicidad**. Es decir, el número de elementos de una clase que participan en una relación. Se puede indicar un número, un rango… Se utiliza *n* o * para identificar un número cualquiera.
- **Nombre de la asociación.** En ocasiones se escriba una indicación de la asociación que ayuda a entender la relación que tienen dos clases. Suelen utilizarse verbos como por ejemplo: “Una empresa contrata a n empleados”



TIPOS DE RELACIONES

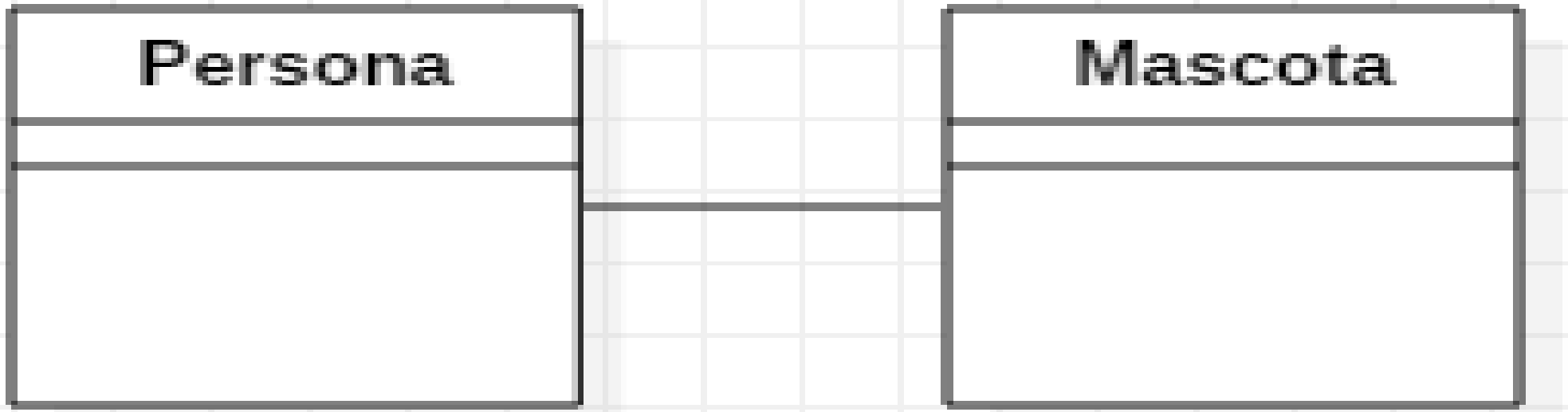
Un diagrama de clases incluye los siguientes tipos de relaciones:

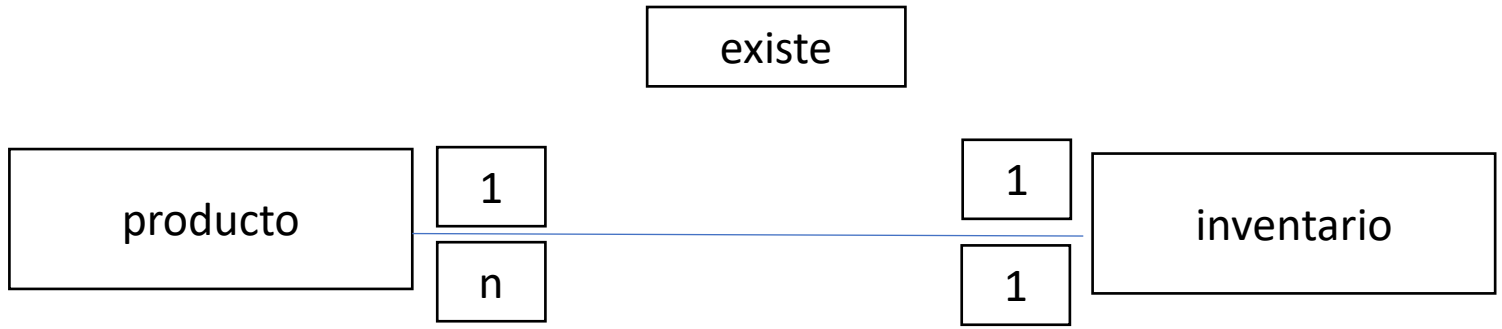
- **Asociación.**
- **Agregación.**
- **Composición.**
- **Herencia.**
- **Dependencia.**



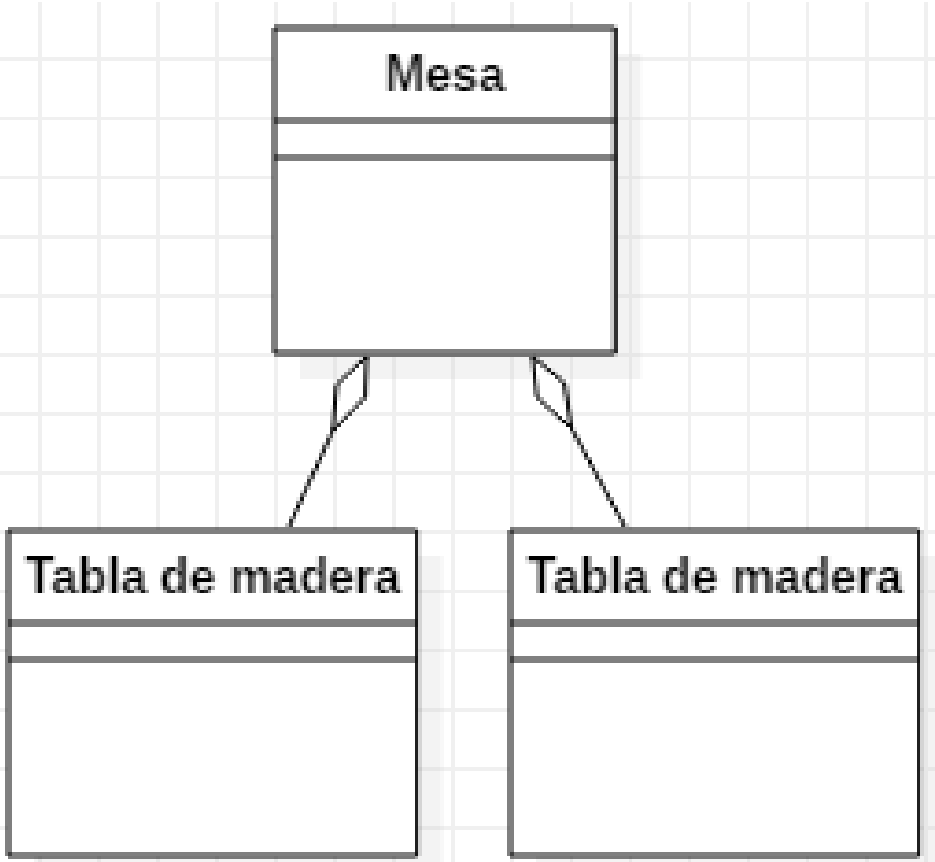
ASOCIACIÓN

- Este tipo de relación es el más común y se utiliza para representar dependencia semántica. Se representa con una simple línea continua que une las clases que están incluidas en la asociación.
- Un ejemplo de asociación podría ser: “Una mascota pertenece a una persona”.
- Multiplicidad




existe
producto
1
n
1
1
inventario

AGREGACIÓN

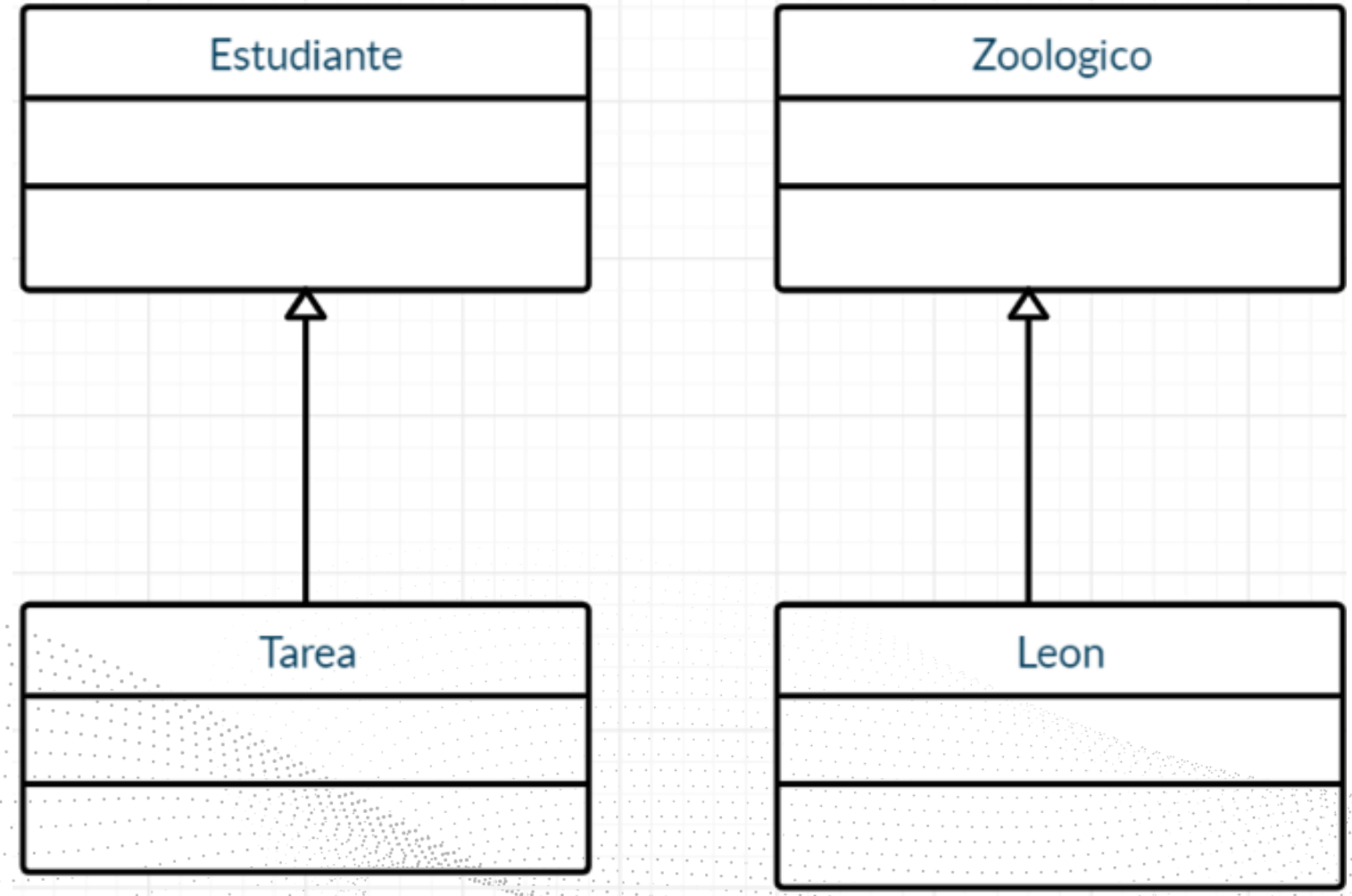


Es una representación jerárquica que indica a un objeto y las partes que componen ese objeto. Es decir, representa relaciones en las que **un objeto es parte de otro**, pero aun así debe tener **existencia en sí mismo**.

Se representa con una línea que tiene un rombo en la parte de la clase que es una agregación de la otra clase (es decir, en la clase que contiene las otras).

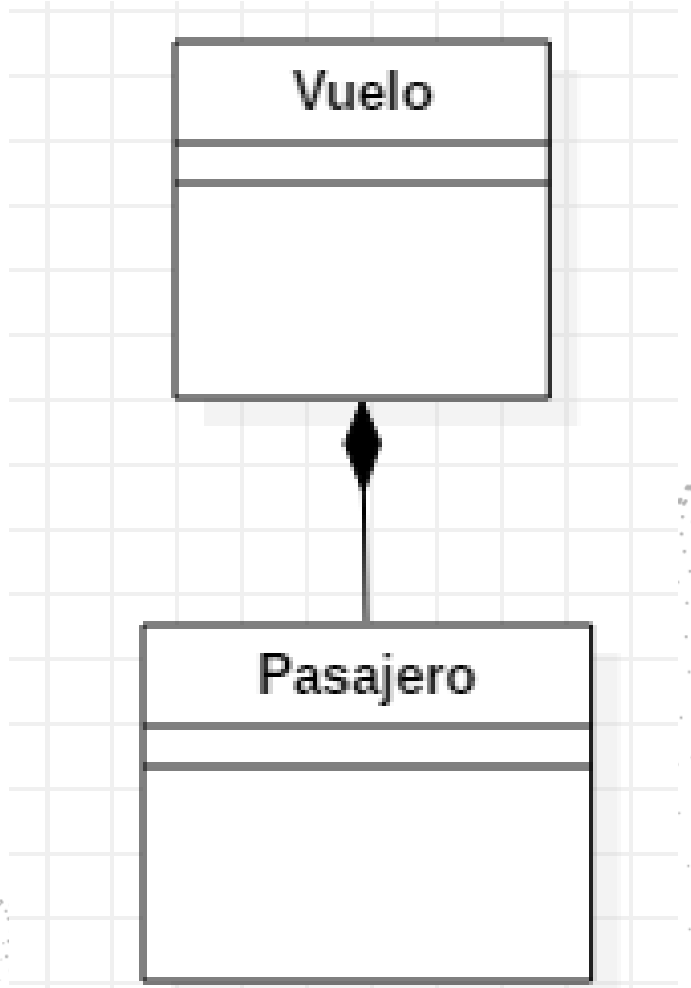
Un ejemplo de esta relación podría ser: “Las mesas están formadas por tablas de madera y tornillos o, dicho de otra manera, los tornillos y las tablas forman parte de una mesa”. Como ves, el tornillo podría formar parte de más objetos, por lo que interesa especialmente su abstracción en otra clase

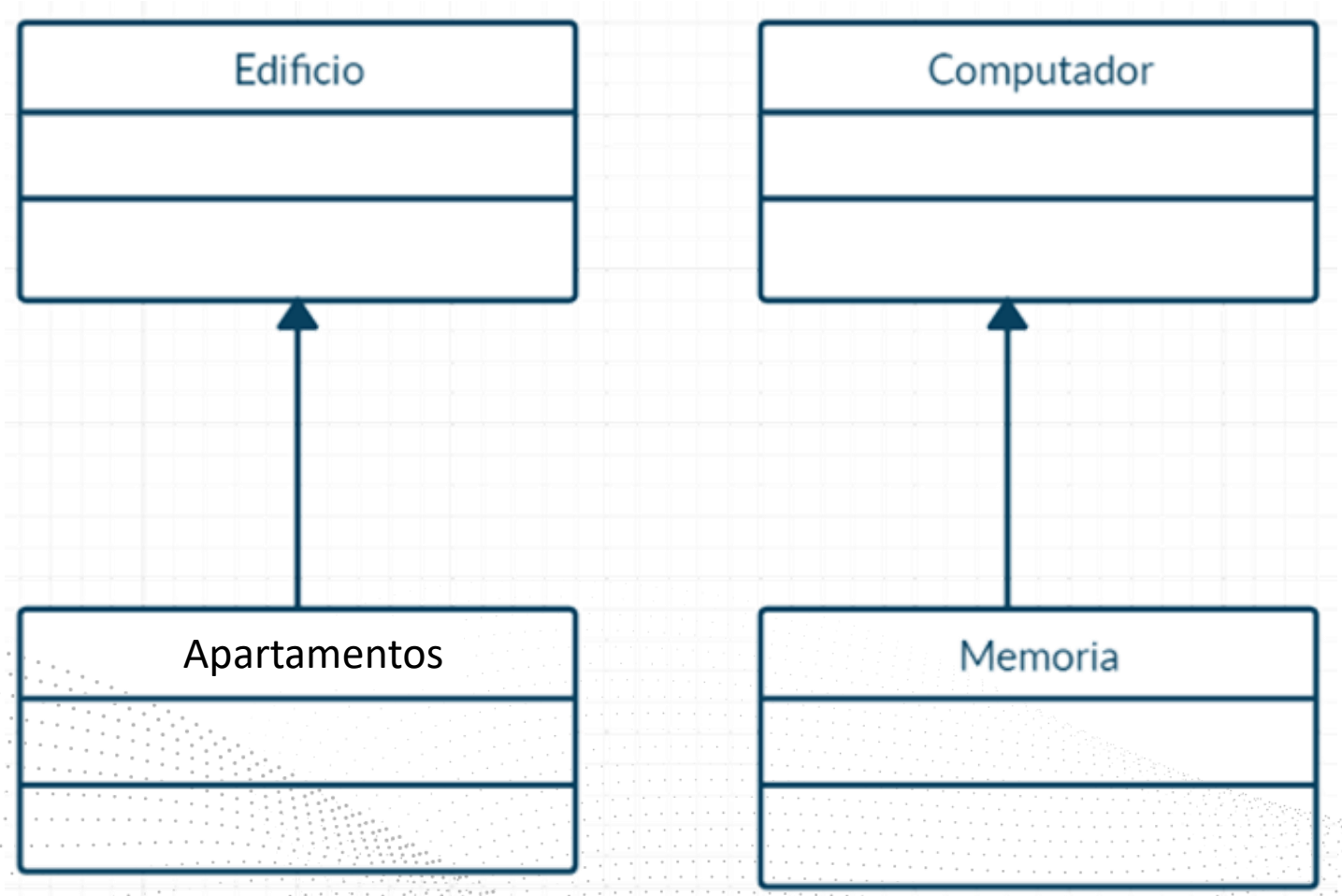
Ejemplos de Agregación



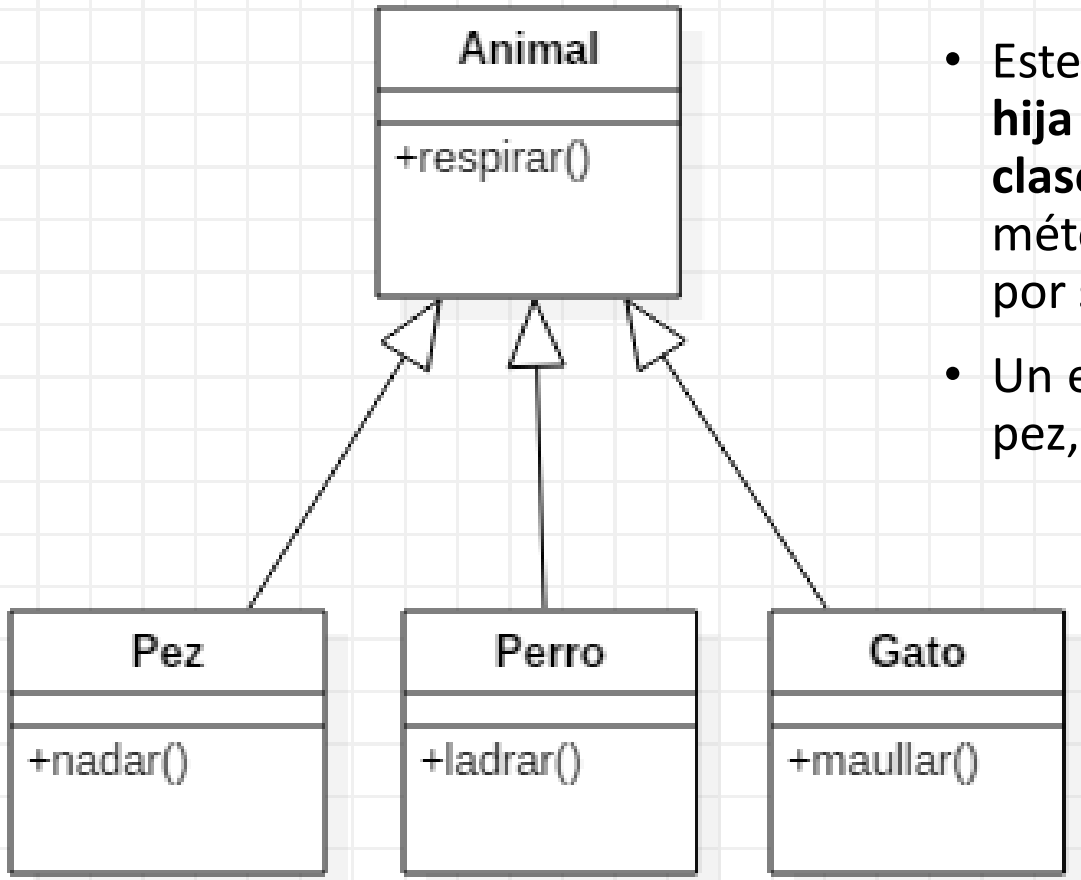
COMPOSICION

- La composición es similar a la agregación, representa una **relación jerárquica entre un objeto y las partes que lo componen**, **pero de una forma más fuerte**. En este caso, los elementos que forman parte no tienen sentido de existencia cuando el primero no existe. Es decir, cuando el elemento que contiene los otros desaparece, deben desaparecer todos ya que no tienen sentido por sí mismos sino que dependen del elemento que componen. Además, suelen tener los mismos tiempo de vida. Los componentes no se comparten entre varios elementos, esta es otra de las diferencias con la agregación.
- Se representa con una línea continua con un rombo relleno en la clase que es compuesta.
- Un ejemplo de esta relación sería: “Un vuelo de una compañía aérea está compuesto por pasajeros, que es lo mismo que decir que un pasajero está asignado a un vuelo”




Edificio
Computador
Apartamentos
Memoria

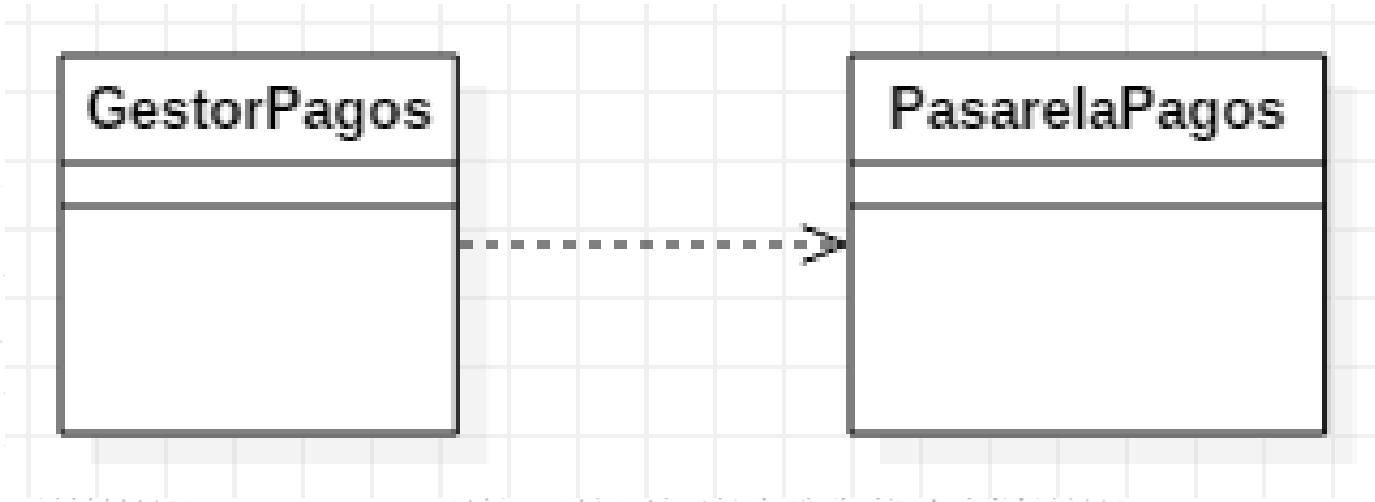
HERENCIA



- Este tipo de relaciones permiten que **una clase (clase hija o subclase) reciba los atributos y métodos de otra clase (clase padre o superclase)**. Estos atributos y métodos recibidos se suman a los que la clase tiene por sí misma. Se utiliza en relaciones “es un”.
- Un ejemplo de esta relación podría ser la siguiente: Un pez, un perro y un gato son animales.

DEPENDENCIA

Se utiliza este tipo de relación para **representar que una clase requiere de otra para ofrecer sus funcionalidades**. Es muy sencilla y se representa con una flecha discontinua que va desde la clase que necesita la utilidad de la otra flecha hasta esta misma.



Gracias