MODELACIÓN DE SISTEMAS DE SOFTWARE

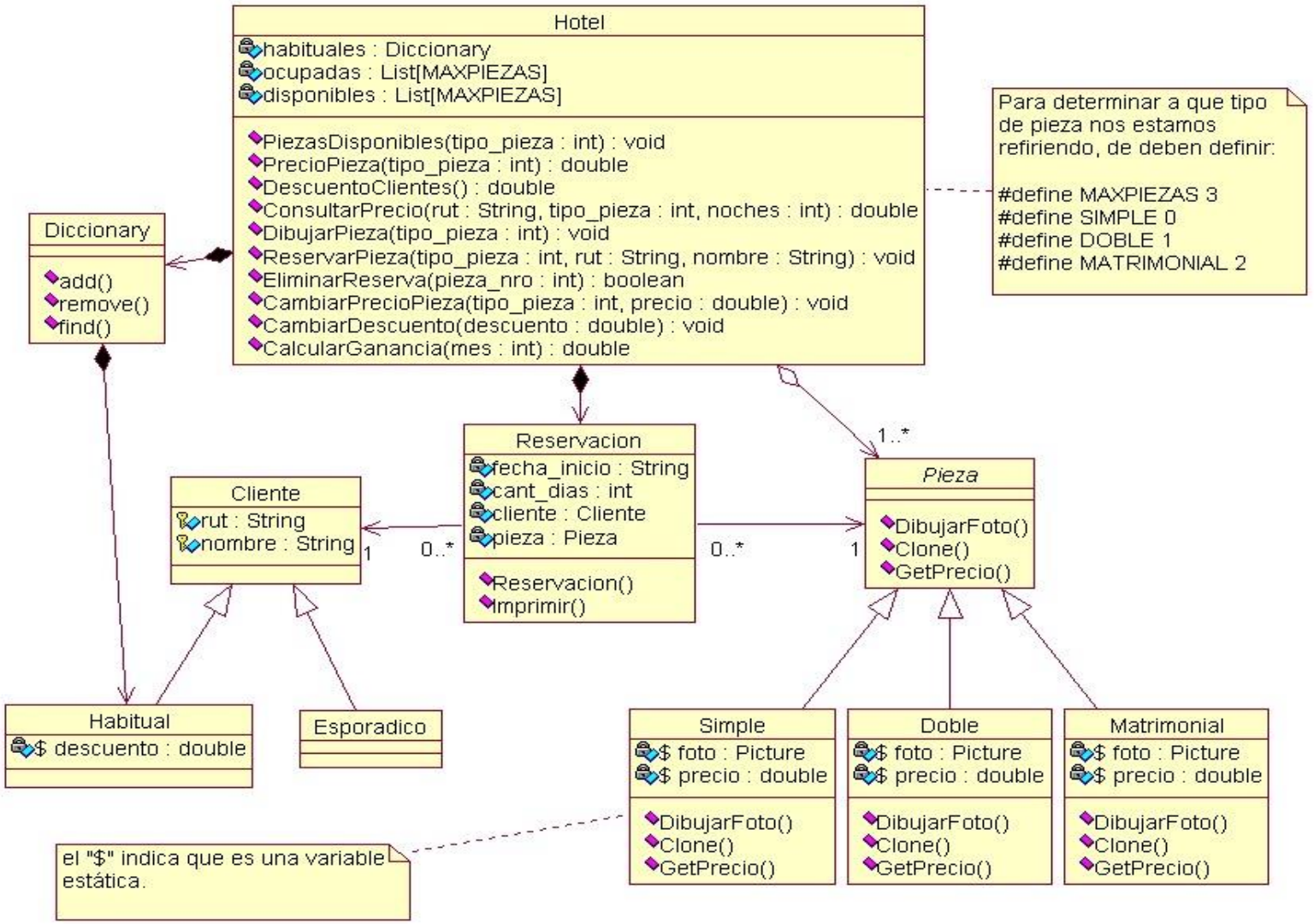
Diagramas de Clases

Diagrama de Clases

- Un diagrama de clases o estructura estática muestra el conjunto de clases y objeto importantes que forman parte de un sistema, junto con las relaciones existentes entre clases y objetos.

- Los diagramas de Clases por definición son estáticos, esto es, representan que partes interactúan entre sí, no lo que ocurre ni cuando ocurre.

Diagrama de Clases



Elementos de un Diagrama de Clases

- Clases
  - Atributos, Métodos
  - Objetos
- Relaciones
  - Herencia
  - Composición,
  - Agregación
  - Asociación

Clases

Las clases describen un conjunto de objetos con propiedades y comportamientos comunes.

Una clase es un constructo que define la estructura y comportamiento de una colección de objeto denominados instancia de la clase.

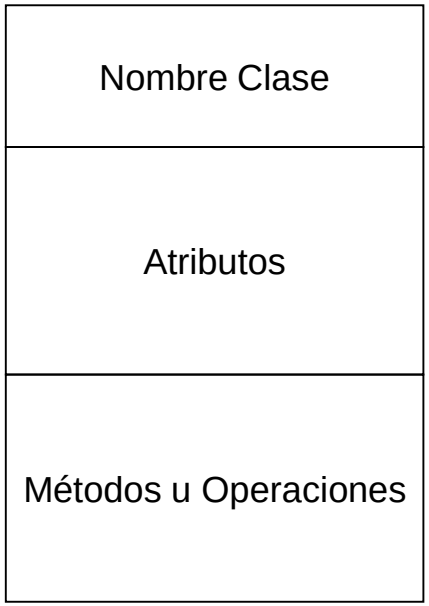
Dentro de la estructura de una clase se definen:

- Atributos. Datos asociados a los elementos y que toman valor al instanciar objetos de una clase.
- Métodos. Funciones o procesos propios de los objetos de una clase

Notación

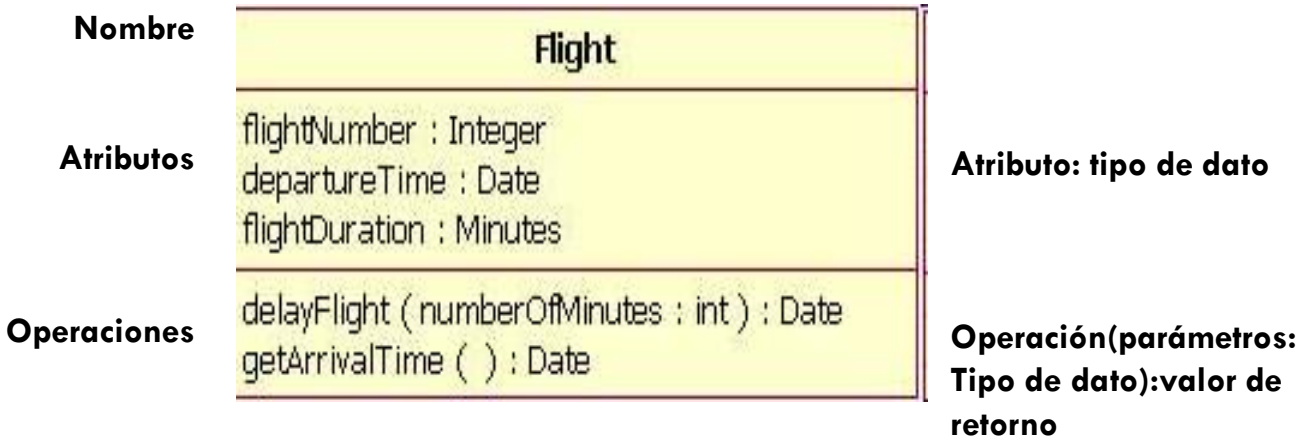
La clase está representada por un rectángulo con tres divisiones internas, son los elementos fundamentales del diagrama.

- El del nombre define la clase, (un tipo de objeto).
- El de los atributos contiene la definición de los datos.
- El de las operaciones contiene la definición de cada comportamiento soportado por este tipo de objeto.



Ejemplo

La siguiente figura muestra un vuelo de una aerolínea modelado como una clase.



Elementos del Diagrama de Clases

- **Atributo:** Representa una propiedad de una entidad. Cada atributo de un objeto tiene un valor que pertenece a un dominio de valores determinado.
- La sintaxis de una atributo es:

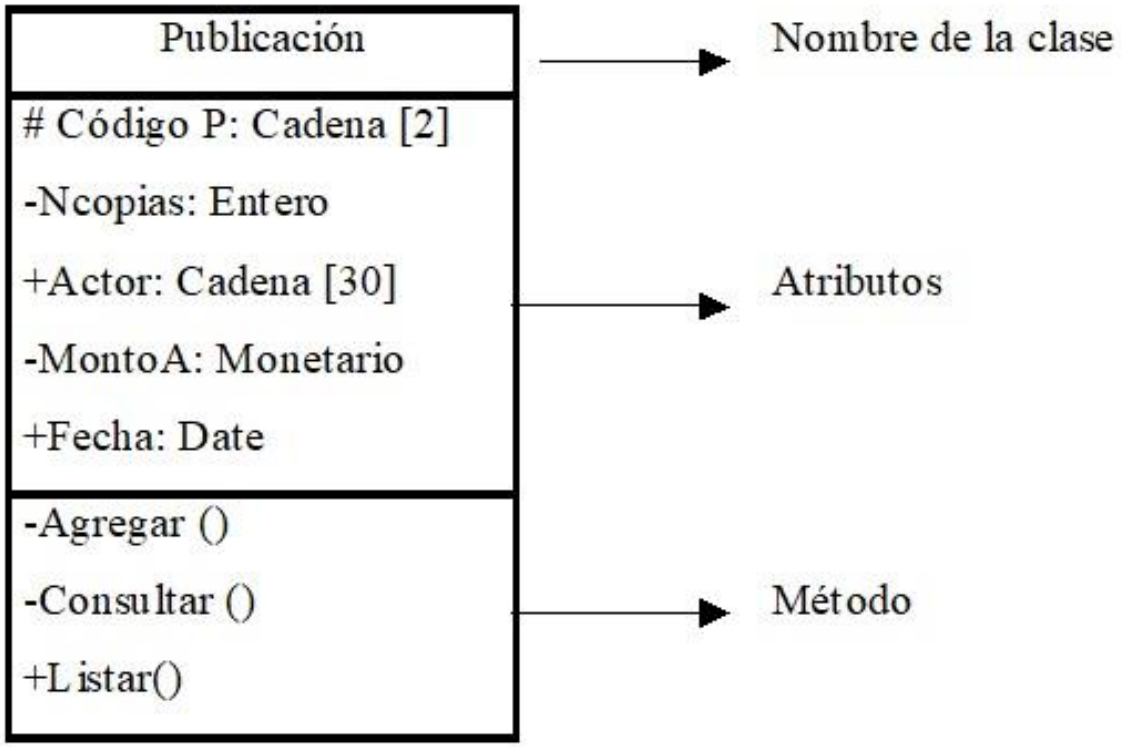
- ***Visibilidad <nombre>: tipo = valor inicial { propiedades}***

Visibilidad de los atributos

La visibilidad puede ser:	Public (+) permite el acceso a objetos de las otras clases.
	Private (-) limita el acceso a la clase, solo operaciones de la clase tienen acceso.
	Protected (#) permite el acceso a subclases. En el caso de generalización (herencia), las subclases deben tener acceso a los atributos y operaciones de la superclase, sino no pueden heredar.
	Package (~) permite el acceso a los otros objetos en el mismo paquete.

Ejemplo

- La siguiente figura muestra una publicación de una biblioteca modelada como una clase:



Elementos del Diagrama de Clases

- **Operación:** El conjunto de operaciones que describen el comportamiento de los objetos de una clase.
- La sintaxis de una operación:

Visibilidad nombre (lista de parámetros): tipo que retorna { propiedades}

Modelando una Operación

Las operaciones requieren un nombre, argumentos y a veces un valor de retorno.

Las reglas de visibilidad se aplican en la misma forma que para los atributos:

- Private,
- Public,
- Protected y
- Package.

Relaciones entre Clases

Las relaciones existentes entre las distintas clases nos indican como se comunican entre sí los objetos de esas clases.

Los mensajes "navegan" por las relaciones existentes entre las distintas clases.

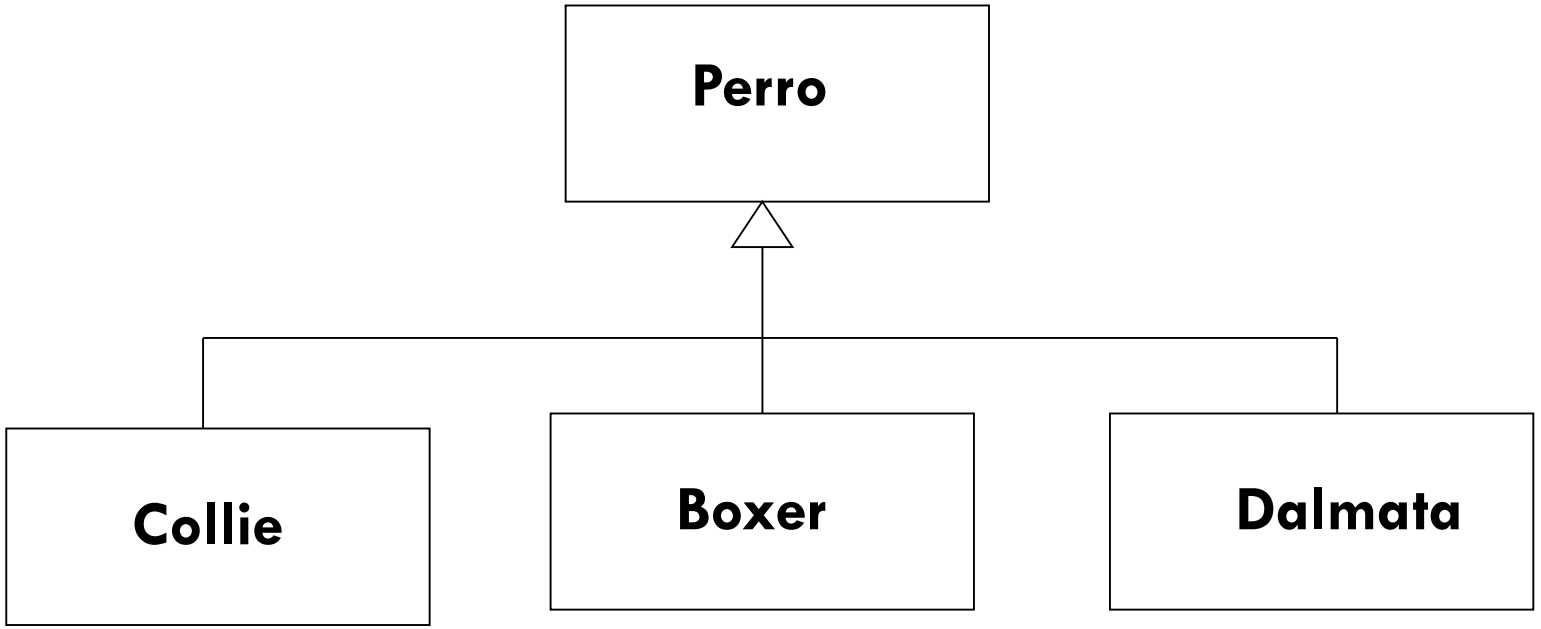
Las relaciones pueden ser:

- Herencia (Generalización),
- Asociación,
- Agregación,
- Composición

Herencia

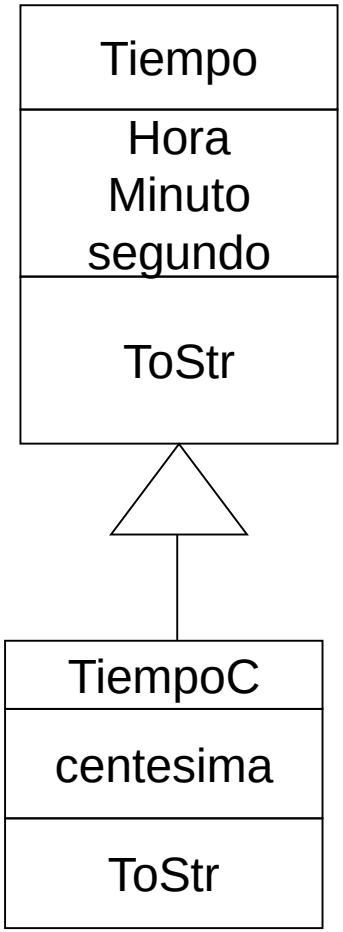
- Indica que una subclase hereda los métodos y atributos especificados por una Super Clase, por ende la Subclase además de poseer sus propios métodos y atributos, poseerá las características y atributos visibles de la Super Clase
- La visibilidad “protected” permite que solo objetos de la misma clase ó subclase vean el elemento.
- La herencia puede ser:
  - Simple,
  - Múltiple

Herencia (Generalización)



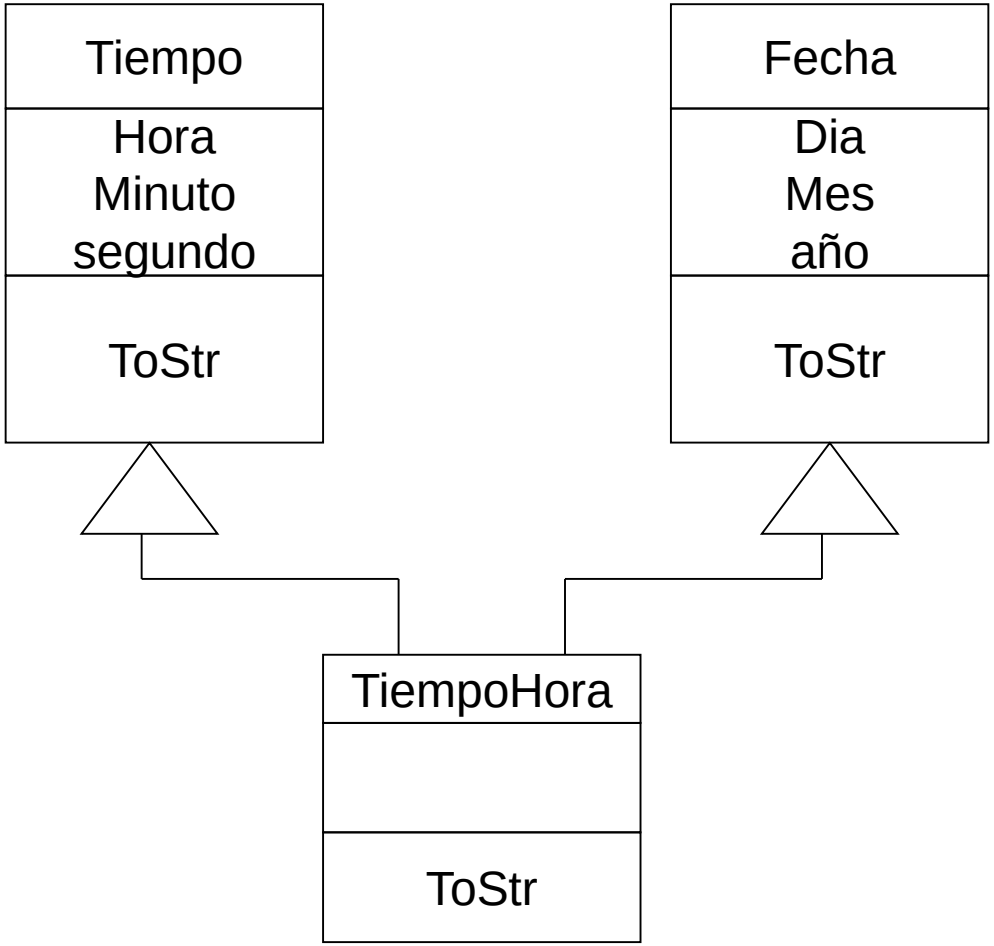
Herencia Simple

- Cuando heredan de una sola clase. Ej.



Herencia Múltiple

- Cuando heredan de dos o mas clases. Ej.



Asociación

- **Una asociación es una relación entre instancias de clases.**
- El propósito de la asociación puede expresarse en un nombre, verbo o frase que describa como los objetos de un tipo (clase) se relacionan con objetos de otro tipo (clase). Por ejemplo:

Una persona **tiene** un coche

Una persona **maneja** un coche

- Multiplicidad: cuantos objetos van a participar en la relación

Asociación

- Las asociaciones se caracterizan por:
- **Rol**. Papel desempeñado por el objeto
- **Multiplicidad**. (Cardinalidad), la cantidad de objetos de una clase que se relacionan con un objeto de la otra clase:

1..*	Uno a muchos
1..n	//
0..*	Cero a muchos
0..n	//
m	Cantidad fija

Elementos del Diagrama de Clases

- La multiplicidad puede ser:

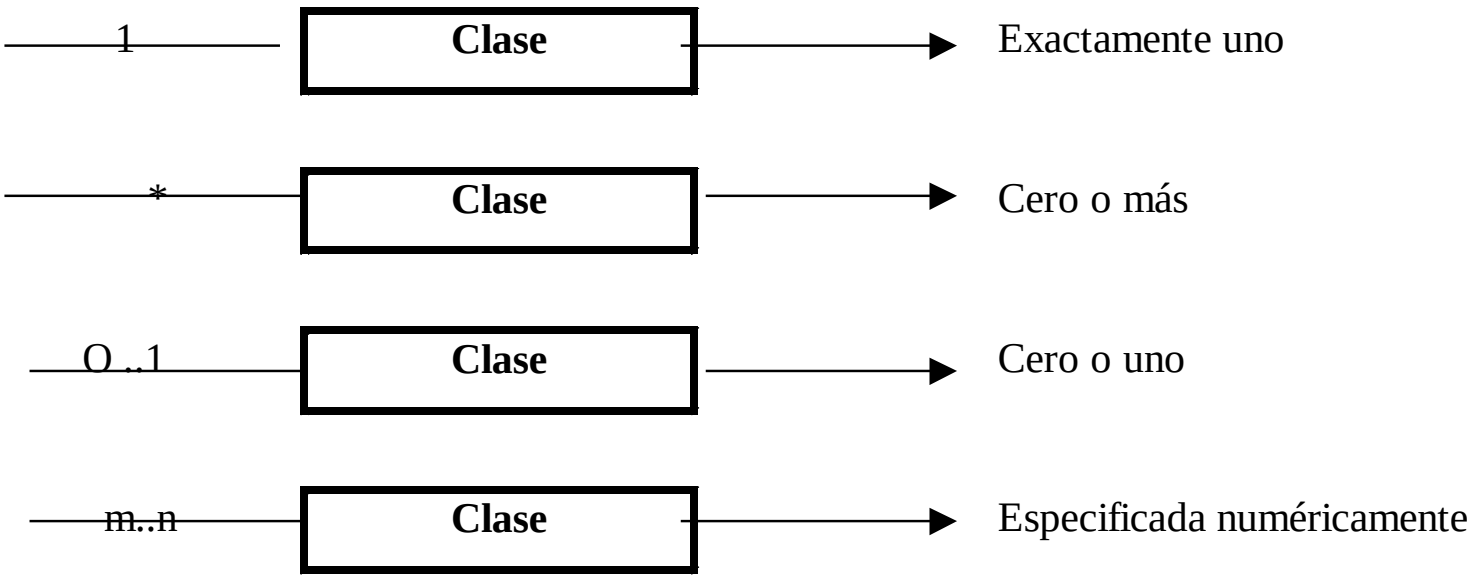


Fig. 3 Tipos de Multiplicidad

Ejemplos de multiplicidad

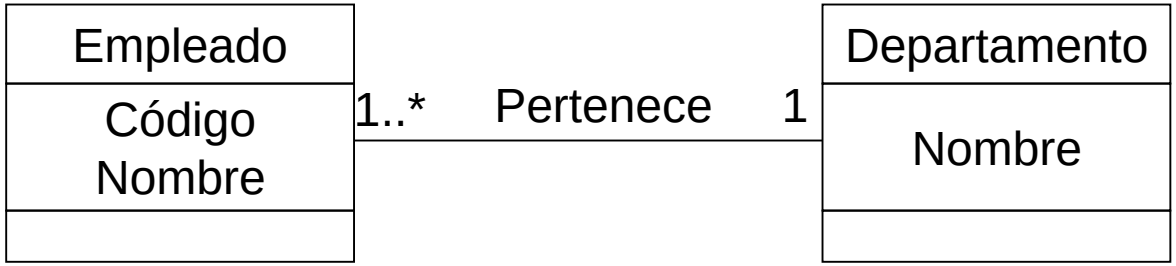
- “Uno a Uno” donde 2 objetos se relacionan de forma exclusiva, uno con el otro
  - P/e Cada universidad tiene un Rector y cada Rector rige una universidad
- “Uno-muchos” donde uno de los objetos puede estar ligado a muchos otros objetos
  - P/e Muchos estudiantes pueden estudiar en una Universidad, y una sola Universidad da estudios a cada Estudiante

Ejemplos de multiplicidad

- “Muchos –muchos” donde cada objeto de cada clase puede estar ligado a muchos otros objetos
  - P/e muchos estudiantes pueden estudiar en varias Universidades

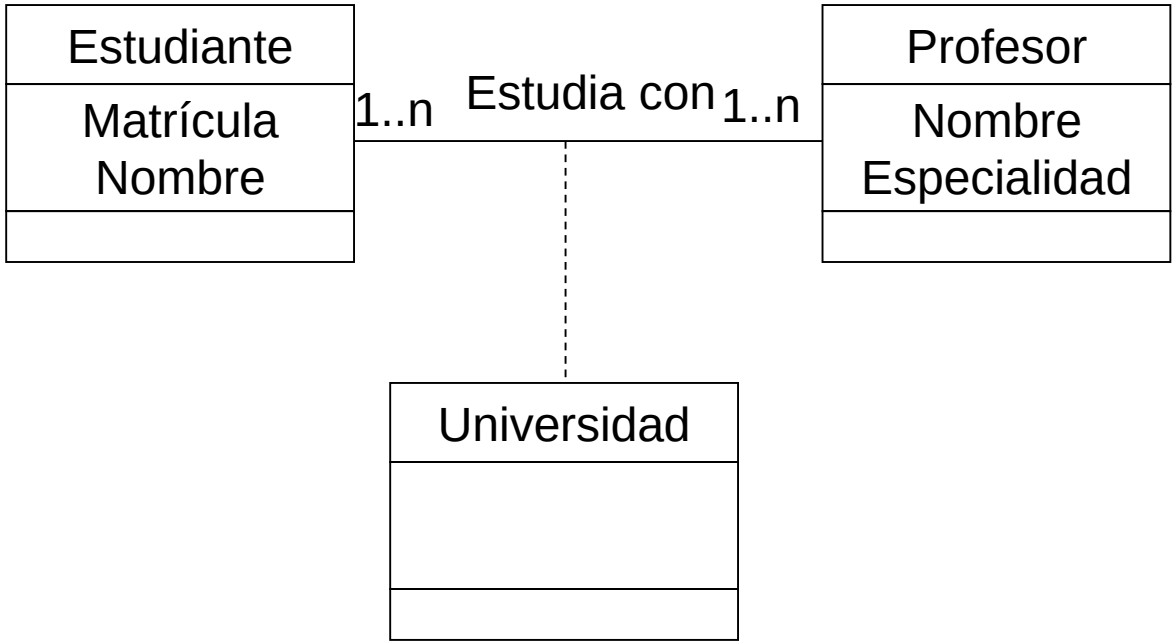
Asociación

Toda carrera tiene un director
Un profesor puede dirigir una carrera

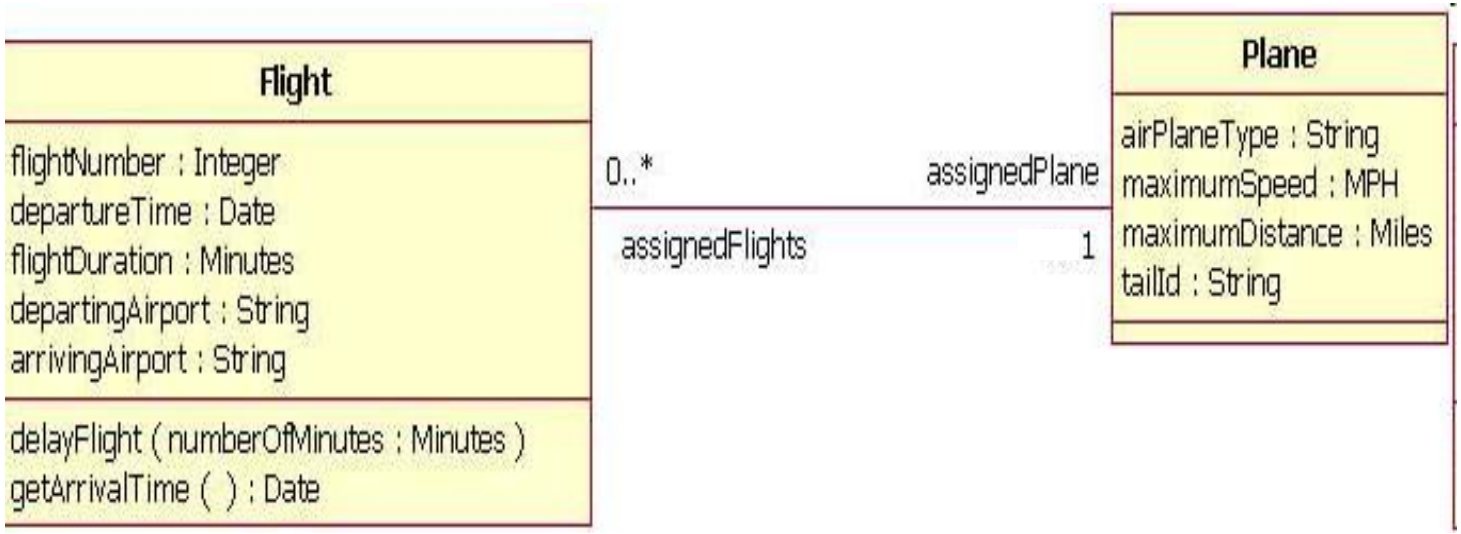


Un empleado pertenece a un departamento
A un departamento le pertenecen 1 o más empleados

Asociación


Estudiante
Matrícula
Nombre
1..n
Estudia con
1..n
Profesor
Nombre
Especialidad
Universidad

Asociaciones



Se indica el rol y la multiplicidad.

Un vuelo está asociado con un avión y un avión puede tener asociados ninguno ó varios números de vuelo.

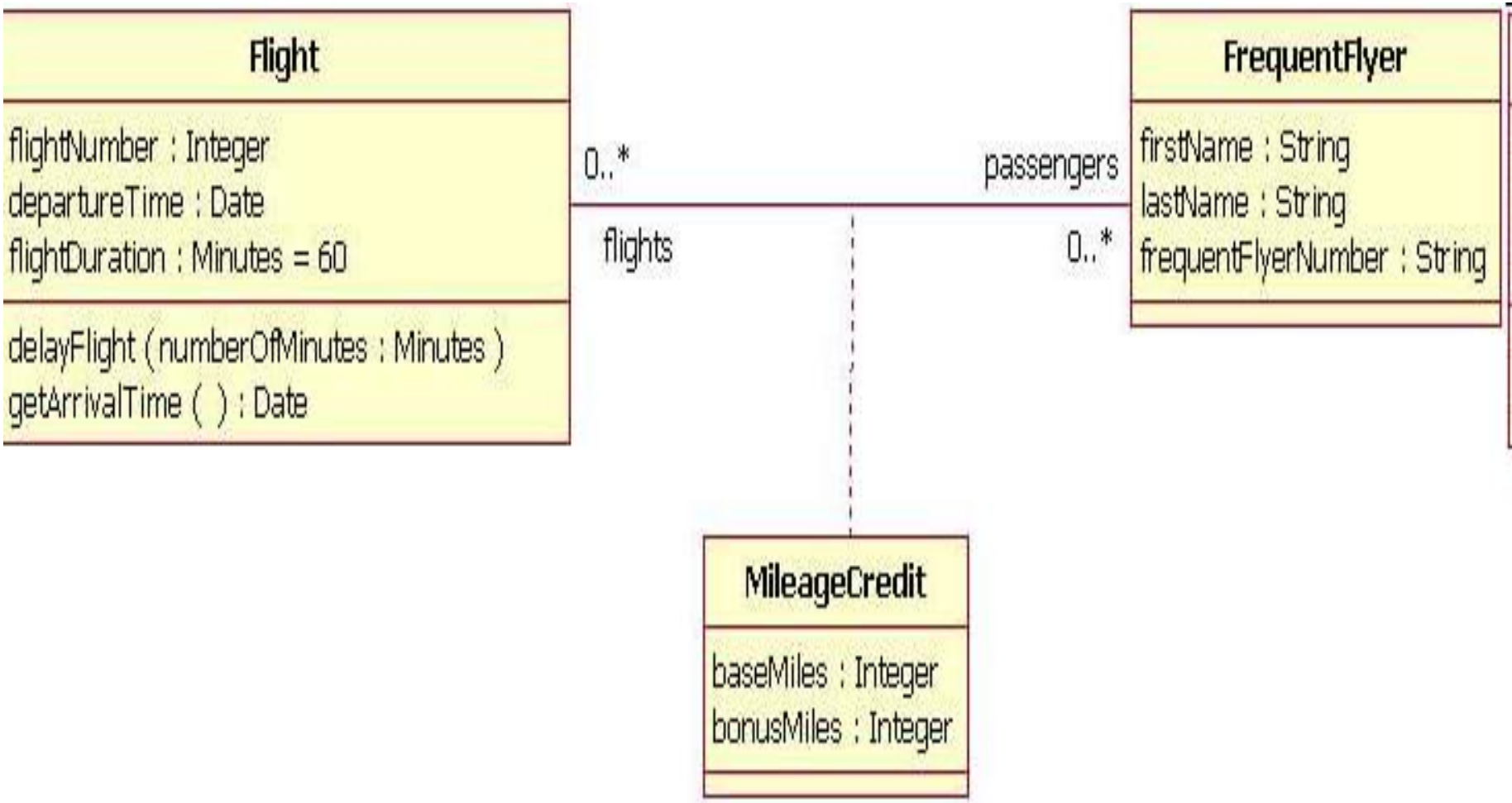
Dirección

- La dirección en las flechas de la asociación determinan en que dirección puede recorrerse una asociación en el momento de la ejecución.
- Una asociación sin flechas significa que se puede ir de un objeto a otro y viceversa.

Clase Asociación

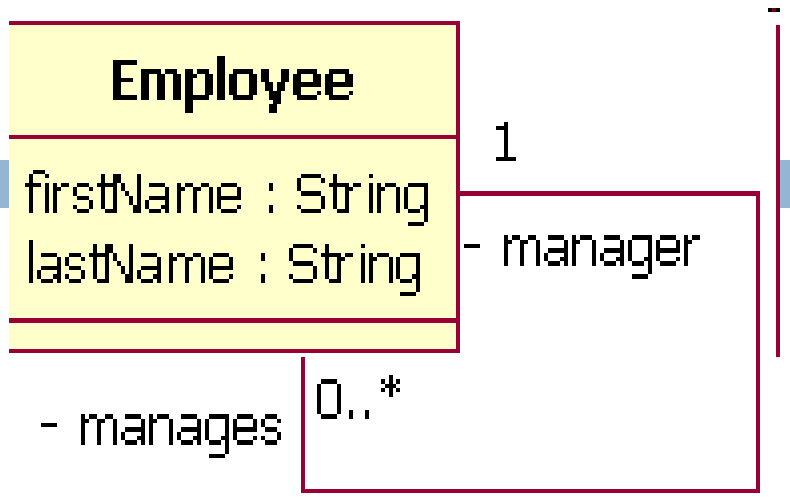
- Cuando se modela una asociación entre clases, a veces es necesario incluir otra clase que contiene información valiosa acerca de la relación.
- Se representa como una clase normal solo que la línea que la une con la línea que conecta las asociaciones primarias es punteada.
- La siguiente figura muestra una clase asociación para el ejemplo de los vuelos.

La asociación entre la clase Flight y FrequentFlyer es a través de una clase llamada MileageCredit. Esto significa que debe haber una instancia en esta clase cuando alguna instancia de la clase Flight se asocie con una instancia de la clase FrequentFlyer



Asociación Reflexiva

- Una clase puede asociarse con sí misma. Una clase Empleado puede relacionarse con sí misma a través del rol gerente/dirige.
- No significa que una instancia está relacionada consigo misma, sino que una instancia de la clase está relacionada con otra instancia de la misma clase.

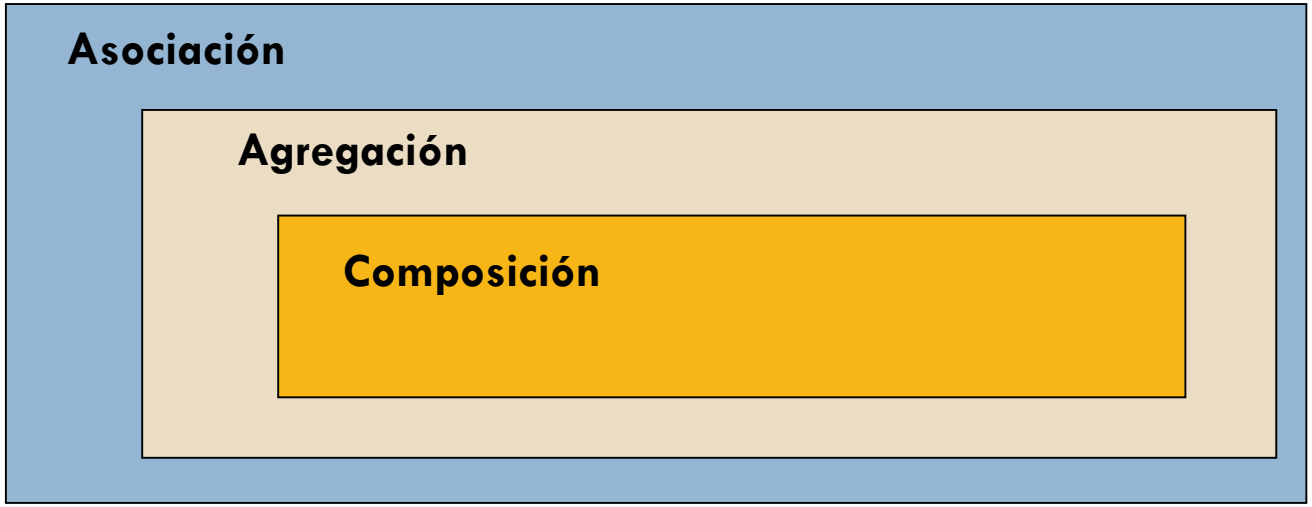


Una instancia de Employee puede ser el gerente de otras instancias de Employee. Como el rol manages tiene una multiplicidad de 0…*, significa que puede no tener otros empleados a quien dirigir.

Una instancia de Employee tiene 1 sólo gerente ó un solo director.

Agregación y Composición

- Cada agregación es un tipo de asociación.
- Cada composición es una forma de agregación.



Agregación básica

Es un tipo especial de asociación utilizado para modelar una relación "whole to its parts".

Por ejemplo, Coche es una entidad "whole" y Llanta es una parte del Coche.

Una asociación con una agregación indica que una clase es parte de otra clase.

En este tipo de asociación, la clase hijo puede sobrevivir sin su clase padre.

Para representar una relación de agregación, se dibuja una línea sólida de la clase padre (total) a la clase hijo (parte), y con un diamante en el lado de la clase padre.
Una llanta puede existir sin automóvil

Agregación

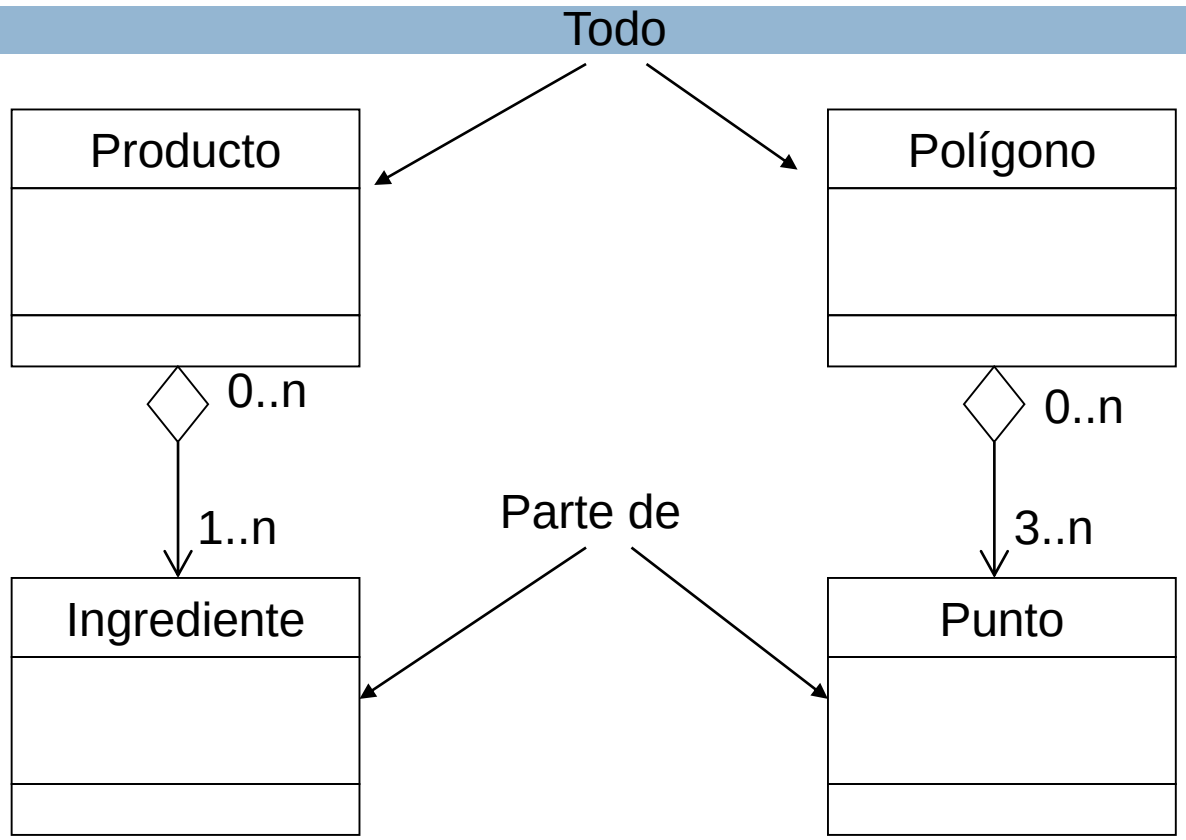
Es una relación de **contenedor** y **contenido,** donde el ***contenedor*** contiene objetos ***contenido***. Se pueden observar las siguientes características:

Independencia existencial: El elemento contenido no desaparece al destruirse el que lo contiene.

Pertenencia débil: Se puede decir que el objeto contenedor no contiene realmente al objeto contenido, sino que tiene una referencia a él.

Compartición: Los objetos contenidos también pueden formar parte del estado de otro objeto

Agregación



Un producto está compuesto por uno o más ingredientes
Un ingrediente puede estar en 0 o más productos.

A un polígono se le puede agregar puntos
Un punto puede formar parte de 0 o más polígonos

Composición

En este caso el ciclo de vida de una instancia de la clase hijo depende del ciclo de vida de una instancia de la clase padre.

A diferencia de la agregación básica, para representarla el diamante no es hueco.

Una instancia de la clase Company debe tener al menos una en la clase Departamento.

En este tipo de relaciones, si la instancia Company se elimina, automáticamente la instancia Departamento también se elimina.

Otra característica importante es que **la clase hijo solo puede relacionarse con una instancia de la clase padre.**

Composición

Composición

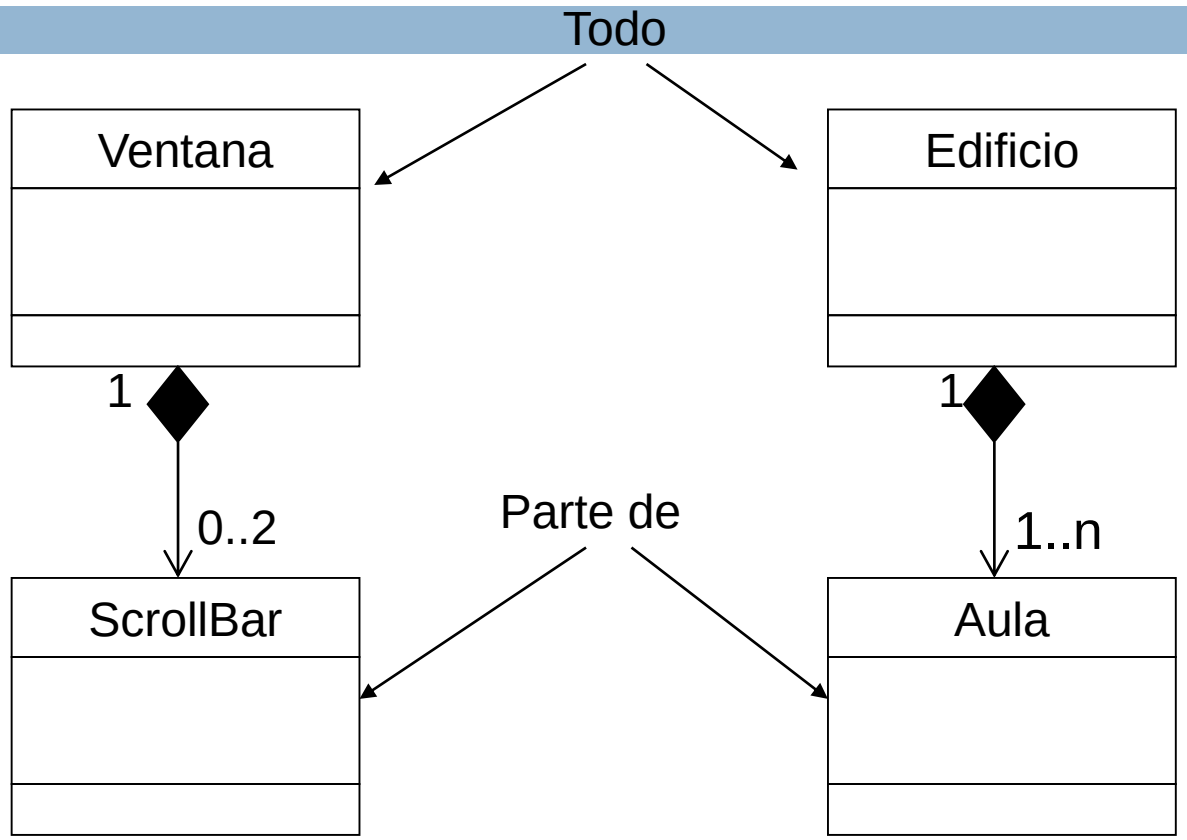
Es una relación de **todo** y **parte de,** donde el ***todo*** esta formado por objetos ***parte de*** que lo componen. Se pueden observar las siguientes características:

Dependencia existencial: El elemento dependiente desaparece al destruirse el que lo contiene y, si es de cardinalidad 1, es creado al mismo tiempo.

Pertenencia fuerte: Se puede decir que el objeto contenido es parte constitutiva y vital del que lo contiene.

No compartición: Los objetos contenidos no son compartidos, esto es, no forman parte del estado de otro objeto

Composición



Una ventana está compuesta por cero o hasta 2 ScrollBars

Un edificio está compuesto por una o más aulas