Diagramas UML

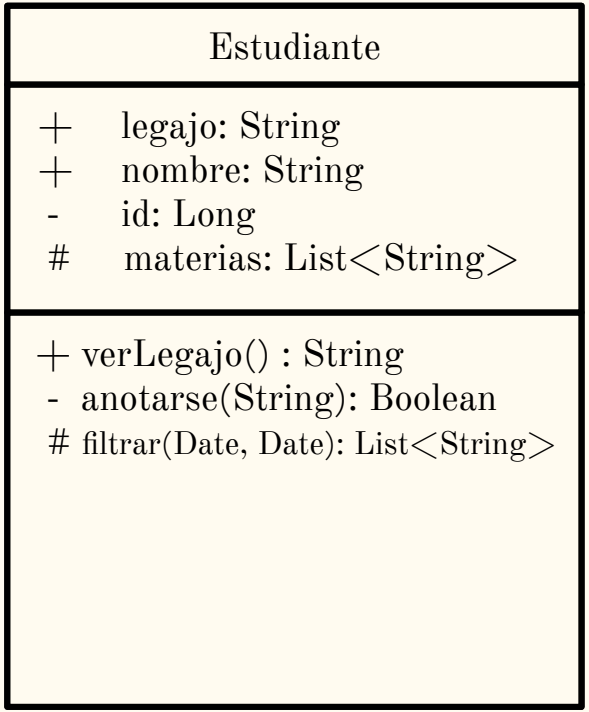
- Que es el diagrama UML? Para qué sirve?
- Principios básicos de los diagramas de clase:
  - Clases
  - Herencia
  - Asociacion, Agregación y Composición
  - Cuantificación
- Principios básicos de los diagramas de secuencia:
  - Clases
  - Loops
  - Llamadas

Qué es UML y para qué sirve?

Los diagramas UML son una representación gráfica de nuestro sistema, o una parte del sistema que estamos queriendo implementar, nos permite mostrar de forma grafica y facil de entender para desarrolladores y no desarrolladores como las distintas partes del sistema interactúan unas con las otras, como estan compuestas y cómo se comunican para llevar a cabo una funcionalidad.

Principios básicos de los diagramas de clases:

Clase:



<-- Toda clase debe tener un nombre

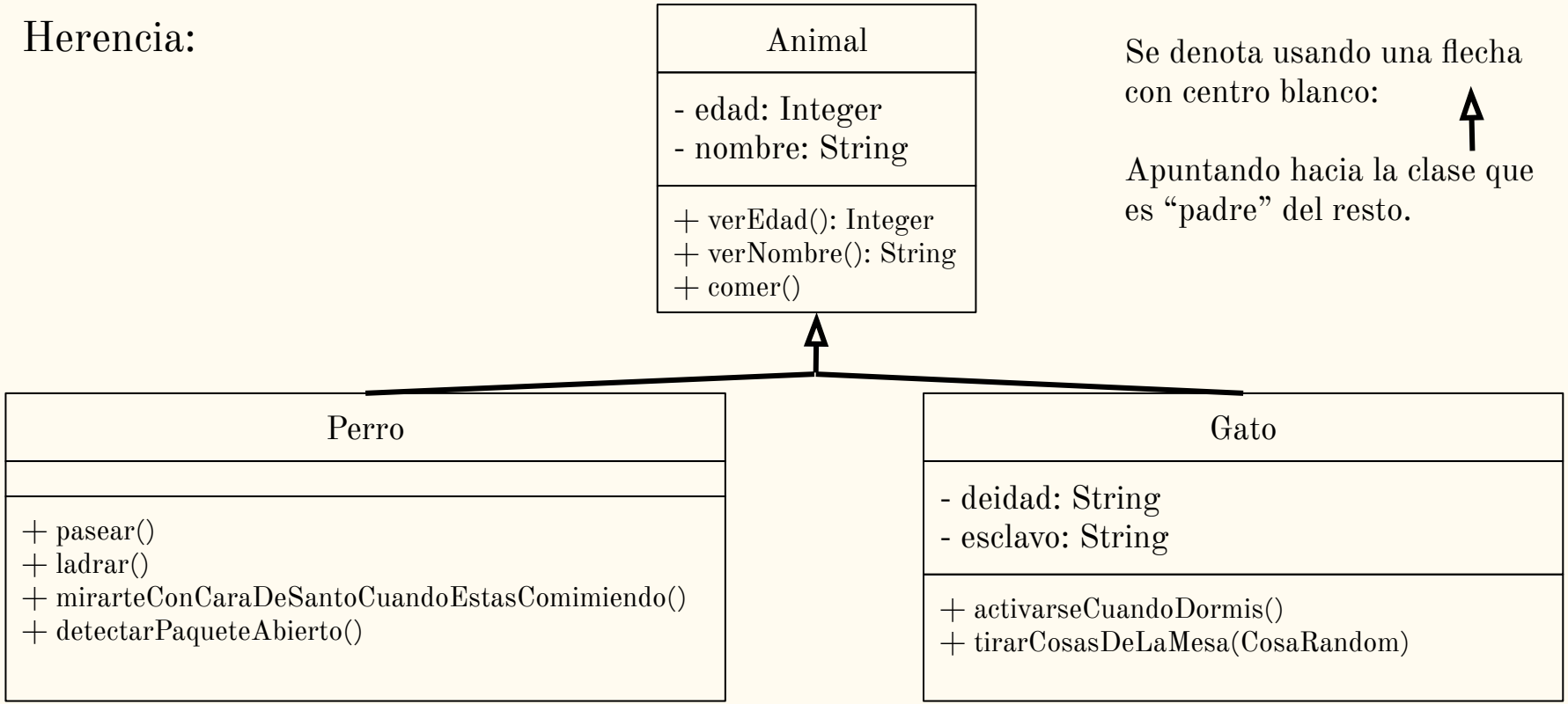
<-- los atributos básicos de nuestra clase, son variables que cambian de valor dependiendo de como se comporta nuestro objeto, generalmente se lo conoce como “estado interno”

+ Significa publico, - privado y # protegido

<-- Son los métodos con los cuales se puede interactuar con nuestra clase, gracias a los métodos podemos cambiar el valor de las variables, y agregarle comportamiento a nuestra clase, al igual que los atributos, estos pueden ser públicos (+), privados (-) o protegidos (#).

Principios básicos de los diagramas de clases:

Herencia:

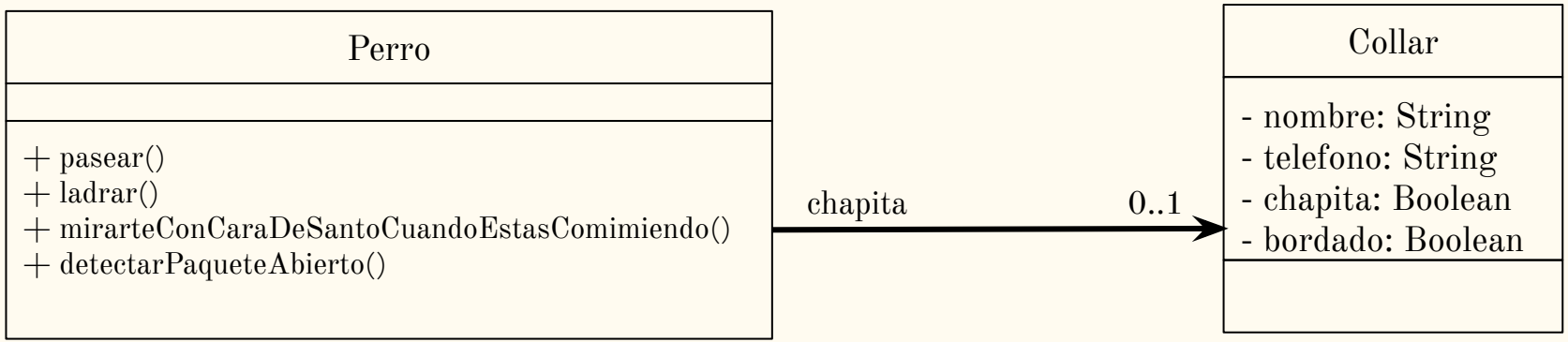


Se denota usando una flecha con centro blanco:

Apuntando hacia la clase que es “padre” del resto.

Principios básicos de los diagramas de clases:

Asociacion:

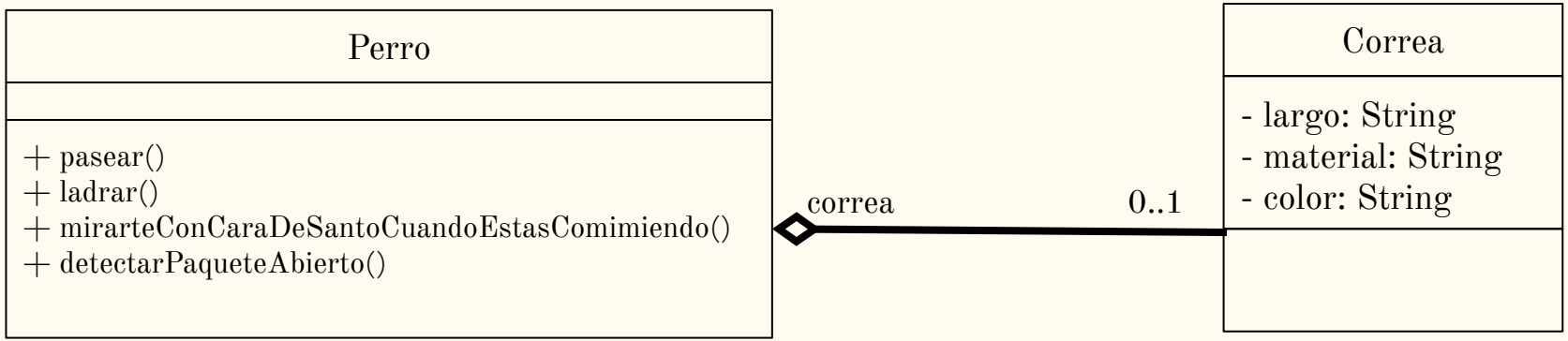


La asociación se denota con una flecha → y significa que el Objeto, en nuestro caso un Perro, posee otro objeto, en el ejemplo es un collar.
El nombre del atributo se pone montado sobre la flecha, pegado al objeto que posee la relación, y del otro lado se coloca la cardinalidad, esta puede ser:

0..1 : se tiene 1 o ningún elemento
0..*: se tiene una lista de elementos, esta puede estar vacía
1: SIEMPRE se tiene una instancia de esa ota clase

Principios básicos de los diagramas de clases:

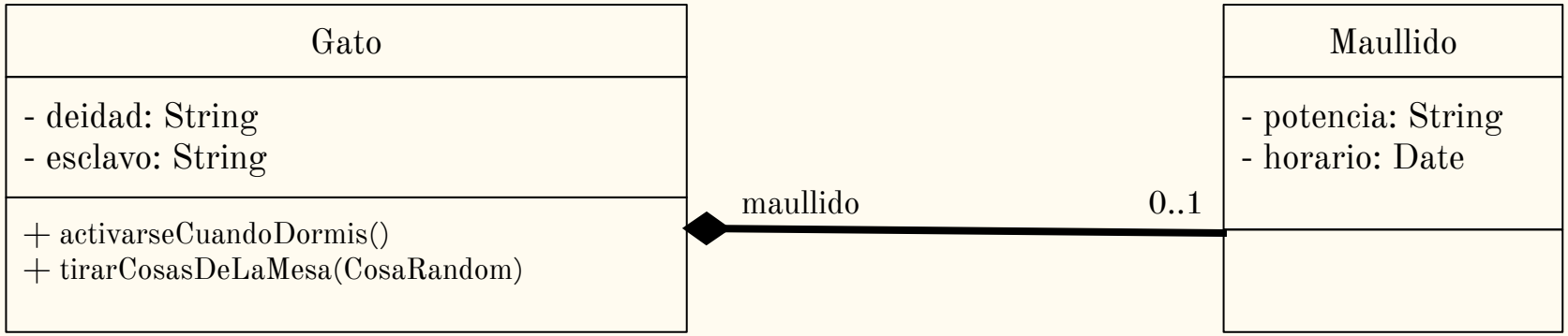
Agregacion:



La agregación se denota con un rombo  y significa que el Objeto más cercano al rombo es el dueño de la agregación, es prácticamente otra forma de anotar asociación, sigue los mismos conceptos, solo que nos da un poco más de información acerca de la relación entre las dos clases, por ejemplo, el Perro puede pasear sin la Correa y la Correa puede ser usada para pasear a más de un mismo Perro, si la correa es destruida el Perro no necesariamente tiene que ser destruido, dado que puede usar otra correa, y lo mismo pasa desde el otro punto de vista.

Principios básicos de los diagramas de clases:

Composicion:



La composición se denota con un rombo lleno:  y significa que el Objeto más cercano al rombo es el dueño de la composición, pero además en caso de desaparecer, también desaparece el objeto de la otra punta del rombo, esto significa que el objeto Maullido del ejemplo NO PUEDE EXISTIR si no está asociado a un Gato. Es otro tipo de asociación, un poco más fuerte que la anterior, pero menos frecuente, la mayoría de las veces las relaciones que construyamos van a ser agregaciones.

Principios básicos de los diagramas de clases:

Cuantificación: Los objetos pueden poseer una cantidad variada de otros objetos, puede poseer uno solo, o ninguno. Esto es importante a la hora de modelar que quede bien indicado en nuestro modelo para evitar confusiones, dado que no es lo mismo tener una lista de Objetos (List<MiClase>) a tener 1 solo objeto en concreto, simplemente una instancia de MiClase. Los casos más comunes son:

0..1: se puede tener 0 o 1 objeto, no mas.

0..*: se tiene un listado de objetos, esta lista puede estar vacía o con N objetos.

1: se tiene siempre 1 objeto es imposible que exista un atributo vacío.

¿Preguntas?