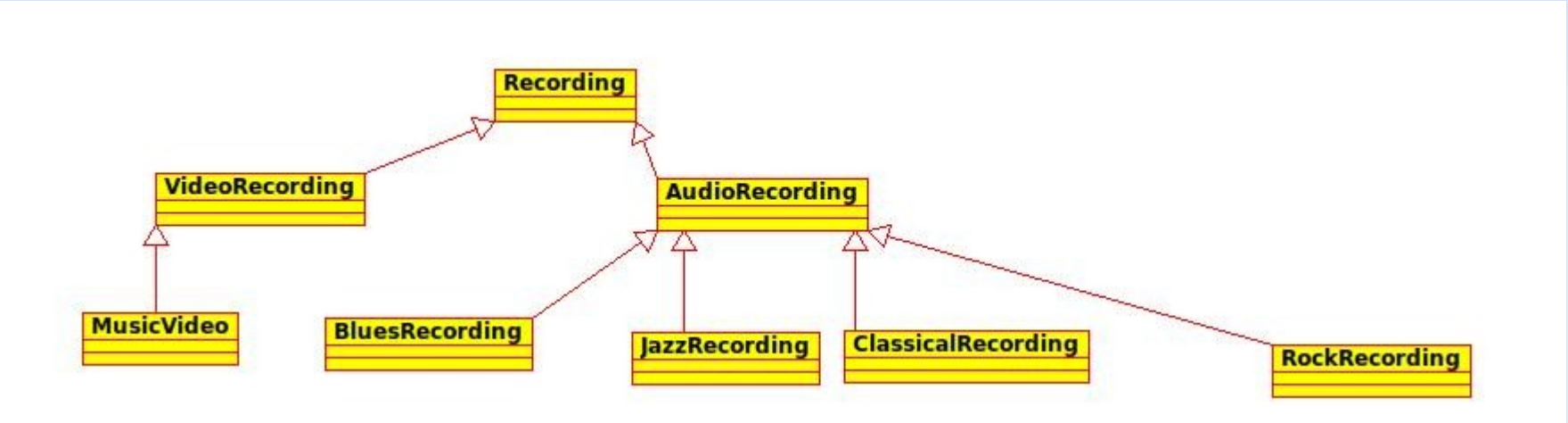
Qué constituye un buen modelo de clases?

“Depende de la aplicación y de los requerimientos que se tengan”

- Evitando especializaciones innecesarias (uso de herencia innecesario)
- Problema: Hacer un modelo para un negocio que vende videos y música.
  La música puede ser jazz, clásica, rock etc**.** Entre los videos existen videos musicales. Al negocio le gustaría poder fácilmente clasificar los aŕticulos que vende por sus categorías (rock, clásica, audio, video, etc)

Modelando la venta de ciertos árticulos

Usar una jeraquía para modelar los artículos que se venden

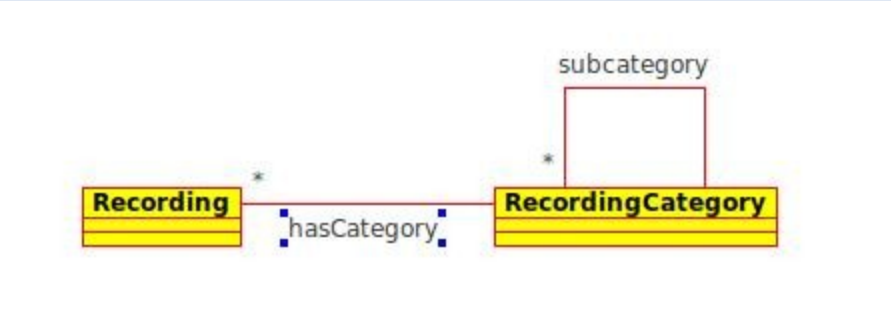


La jerarquía propuesta satisface el concepto de subtipos y podría usarse.
Pero, para justificar realmente una jerarquía debe existir tambien alguna operación (método) que debe ser implementado de manera distinta en las subclases

Modelando la venta de ciertos artículos

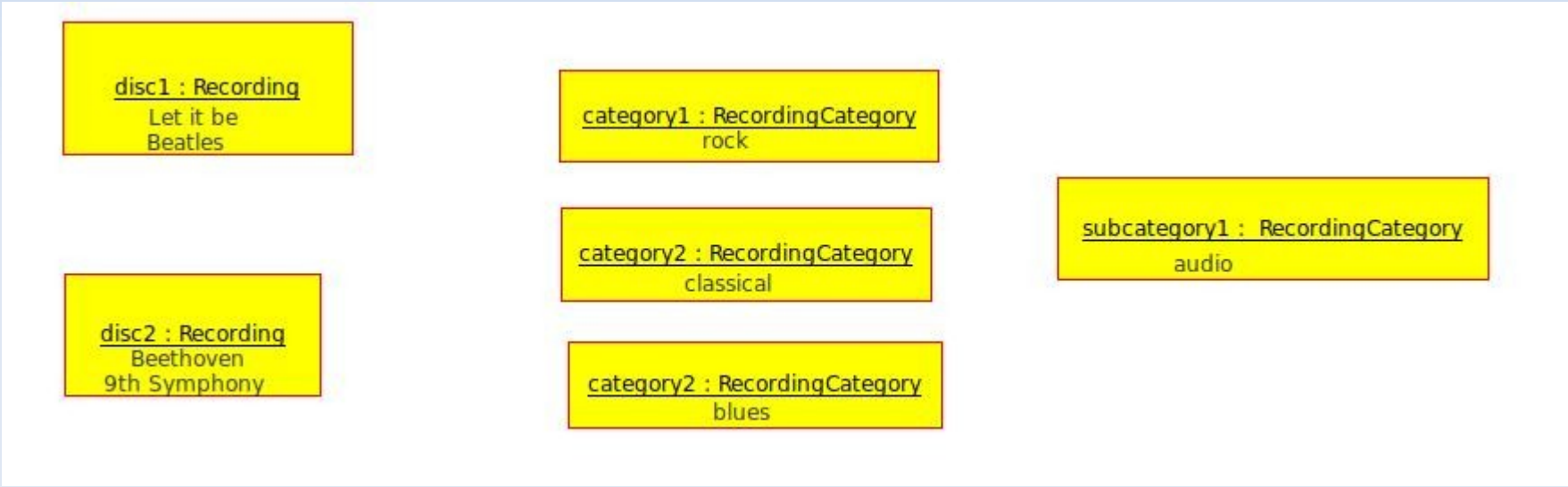
Qué métodos distintos podemos tener para las clases JazzRecording y ClassicalRecording? Ellas no se diferencian en la forma en que se venden.

Propuesta alternativa:



Modelando la venta de ciertos artículos

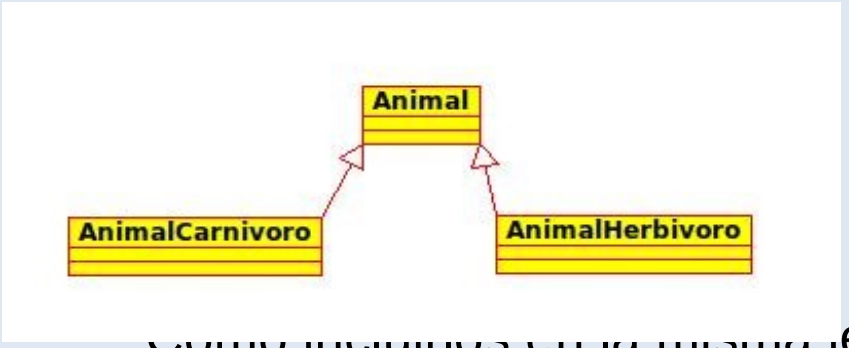
Cómo se ven ciertas instancias (objetos) de las clases anteriores?

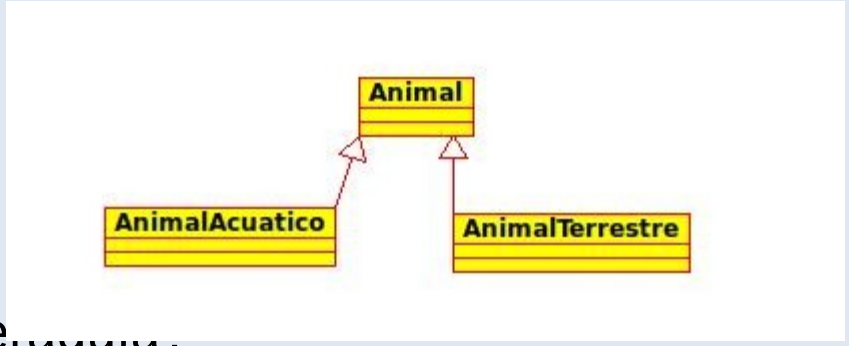


Nota: Falta agregar los conectores. Que objetos están conectados?

Modelando objetos con múltiples discriminadores

- Problema: Cuál será la mejor manera de modelar animales los cuales se quiere poder clasificar segun su tipo de alimentación (carnívoros, herbívoros y omnívoros), y al mismo tiempo según su habitat (acuáticos y terrestres)?
- Primera Propuesta:

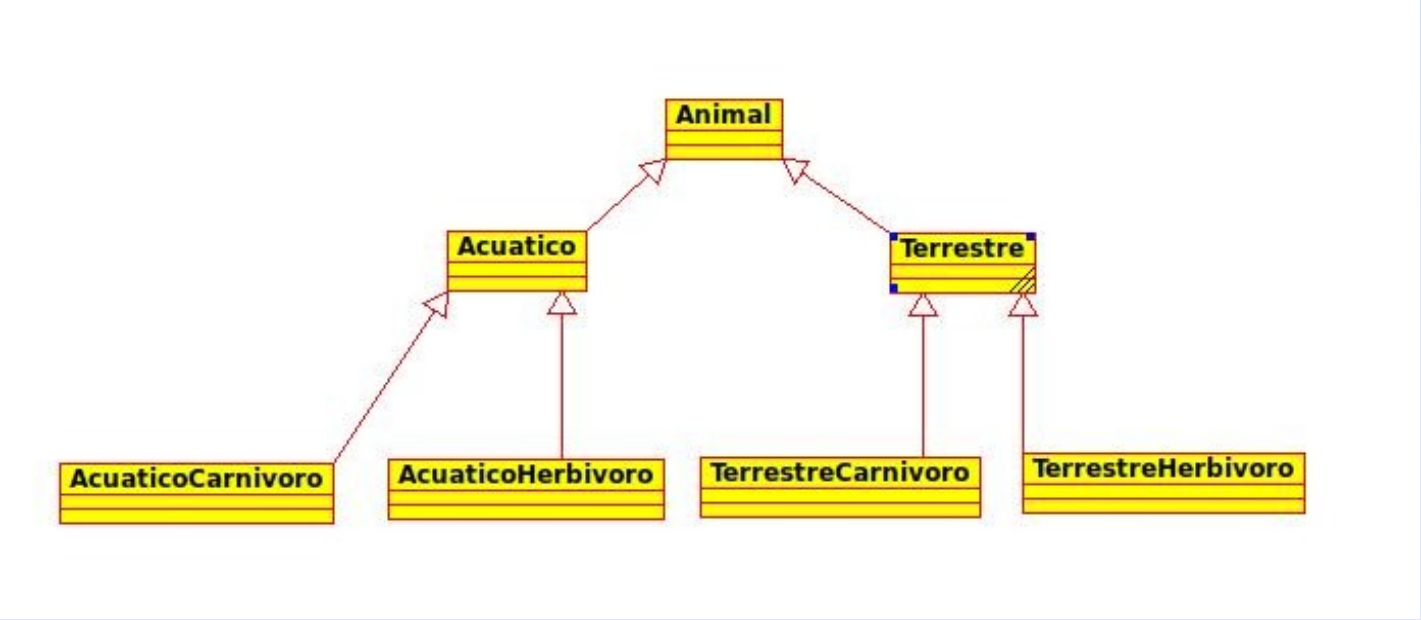




Como incluirlos en la misma jerarquía?

Ejemplo: modelando animales

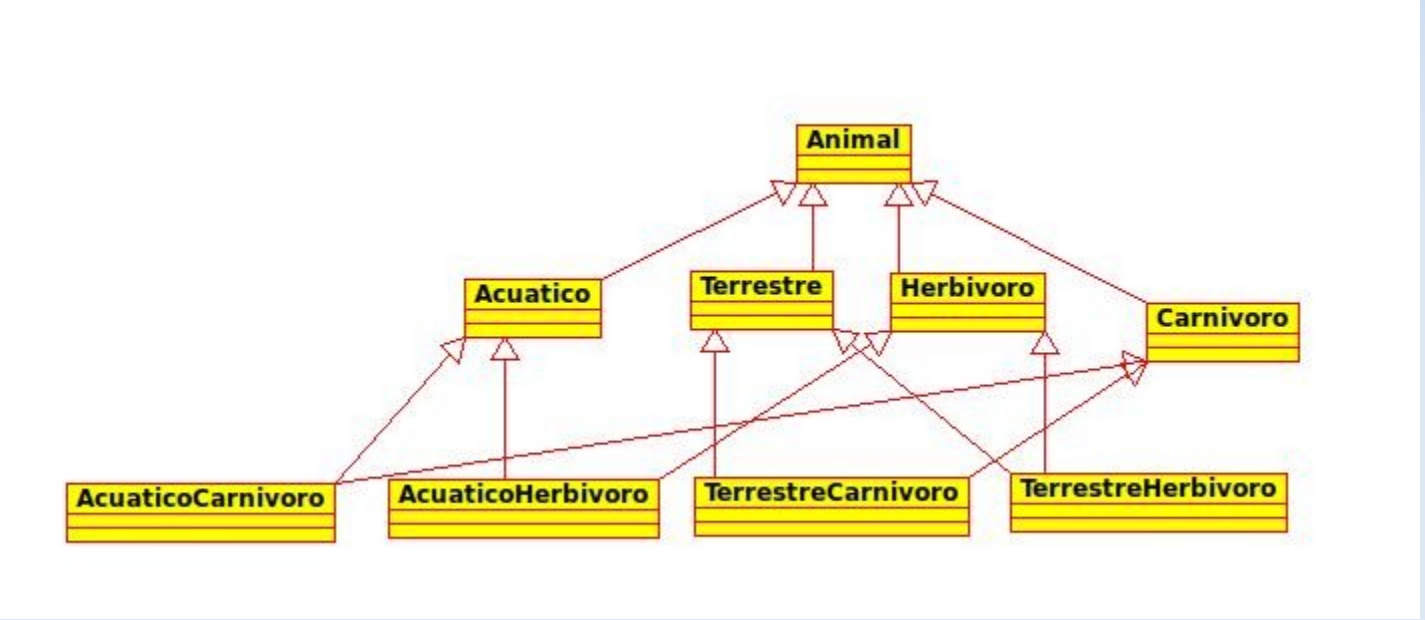
Segunda propuesta:



Dónde están los omnívoros? El diagrama debe crecer en dos clases más

Ejemplo: modelando animales

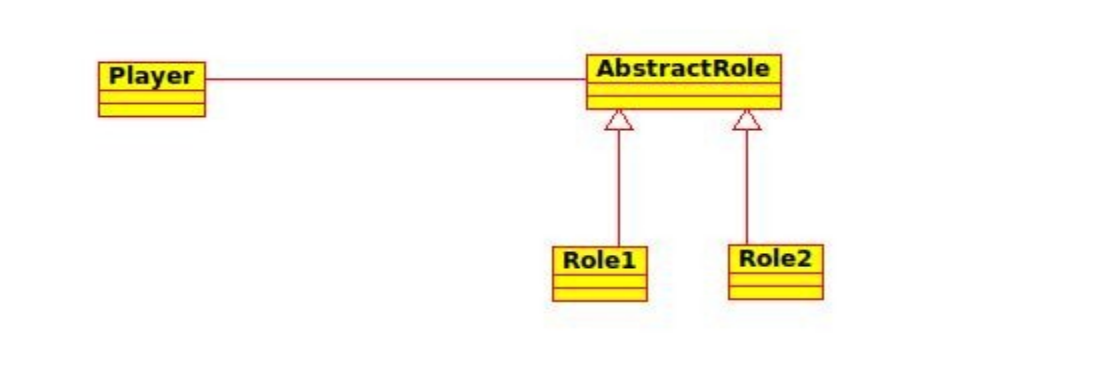
Tercera propuesta: usando herencia múltiple?



Esta solución evita duplicación pero usa aun mas clases y es más compleja que la anterior. Habrá una solución mejor?

Patrón de diseño Player-Role

- En qué consiste?
- Rol: es un conjunto particular de propiedades asociadas con un objeto en un contexto particular.
- Un objeto puede jugar diferentes roles en diferentes contextos.



- Es deseable manejar la encapsulación capturando la información asociada a cada rol en una clase. La multiplicidad puede ser 1 es a 1 o 1 es a muchos.

Modelando estudiantes

- Por ejemplo: un estudiante puede jugar un rol de graduado o no graduado en algún momento. Tambien puede estar jornada parcial o jornada completa.

Cómo modelar los animales?

- Jugador: Animal (carnívoro, herbívoro y omnivoro)
- Rol: Habitat (acuático, terrestre,..)

Antipatrones (Antipatterns)

- Antipatrones: son soluciones de inferior calidad o que no funcionan en un contexto dado
- Ejemplo: un antipatrón de player-rol pattern sería mezclar todos los roles y los atributos del jugador en una sola clase. Esto crea una clase sobrecargada y menos reusable. Al mismo tiempo se pierde el poder de la orientación a objetos.