

UNIDADE DIDACTICA 2: ELABORACIÓN DE DIAGRAMAS DE COMPORTAMIENTO

1. El origen del UML: Unified Modeling Language

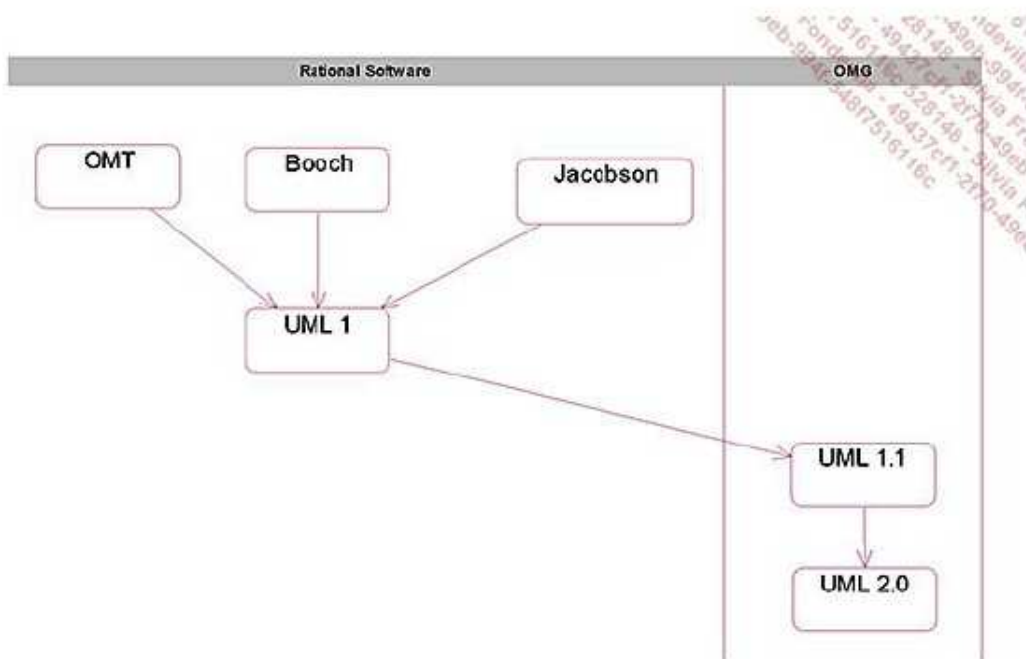
El UML está basado en la orientación a objetos, sistema que vio la luz mucho antes que el UML en el campo de los lenguajes de programación. Simula, el primer lenguaje orientado a objetos nació en los años 1960 y conoció numerosos sucesores: Smalltalk, C++, Java o, más recientemente, C#.

En un lenguaje de programación la descripción de los objetos se realiza de manera formal con una sintaxis rigurosa. Dicha sintaxis resulta ilegible para los no programadores y difícil de descifrar para los programadores. A diferencia de las máquinas, los humanos prefieren utilizar lenguajes gráficos para representar abstracciones, ya que dominan este tipo de lenguaje con mayor facilidad y obtienen una visión de conjunto de los sistemas en mucho menos tiempo.

En los años 1980 y principios de los 1990, las notaciones gráficas se multiplican y, muy a menudo, cada uno utiliza su propia notación. En 1994, James Rumbaugh y Grady Booch deciden unirse para unificar sus notaciones, procedentes de sus respectivos métodos: OMT para James Rumbaugh y método Booch para Grady Booch. En 1995, Yvar Jacobson decide unirse al equipo de los "tres amigos". El equipo trabaja entonces dentro de Rational Software.

La versión 1.0 de UML se publica en 1997. El trabajo de evolución de la notación empieza a hacerse demasiado voluminoso para sólo tres personas y los tres amigos solicitan la ayuda del Object Management Group (OMG), un consorcio de más de 800 sociedades y universidades que trabajan en el campo de las tecnologías del objeto. La OMG adopta la notación UML en noviembre de 1997 en su versión 1.1

La versión 2 de UML, objeto de la presente obra, se encuentra en la actualidad en su forma definitiva. Constituye la primera evolución importante desde la aparición del UML en 1997. A ella se han añadido numerosos diagramas y los ya existentes se han enriquecido con nuevas construcciones.



Origen y principales versiones de UML

Recordemos que UML es una notación destinada al modelado de sistemas y de procesos mediante objetos. UML constituye un soporte de modelado.

2. Conceptos y principios de la orientación a objetos en los que se basa el lenguaje UML.

En primer lugar, trataremos el concepto de objeto y después veremos cómo modelarlo en UML por abstracción.

Introduciremos la noción de clases, representación común de un conjunto de objetos similares.

Hablaremos después del principio de encapsulación, ocultación de informaciones internas y propias del funcionamiento del objeto.

Describiremos las relaciones de especialización y de generalización que introducen las jerarquías de clases, la herencia, las clases concretas y abstractas y, posteriormente, abordaremos el polimorfismo, consecuencia directa de la especialización.

Finalmente trataremos la composición de objetos para concluir con una noción más específica de UML, la especialización de los elementos del diagrama a través de los estereotipos.

2.1. El objeto

Un objeto es una entidad identificable del mundo real. Puede tener una existencia física (un caballo, un libro) o no tenerla (un texto de ley). Identificable significa que el objeto se puede designar.

Ejemplo

Mi yegua Jorgelina
Mi libro sobre UML
El artículo 293B del código de impuestos

Todo sistema concebido en UML está compuesto por objetos que interactúan entre sí y realizan operaciones propias de su comportamiento.

En UML todo objeto posee un conjunto de atributos (estructura) y un conjunto de métodos (comportamiento). Un atributo es una variable destinada a recibir un valor. Un método es un conjunto de instrucciones que toman unos valores de entrada y modifican los valores de los atributos o producen un resultado.

Ejemplo

Una manada de caballos es un sistema de objetos que interactúa entre sí, cada objeto posee su propio comportamiento.

Incluso los objetos estáticos del mundo real son percibidos siempre como dinámicos. Así, en UML, un libro se percibe como un objeto capaz de abrirse él mismo en una página determinada.

2.2. La abstracción

La abstracción es un principio muy importante en modelado. Consiste en tener en cuenta únicamente las propiedades pertinentes de un objeto para un problema concreto. Los objetos utilizados en UML son abstracciones del mundo real.

Ejemplo

Si nos interesamos por los caballos en su actividad de carrera, las propiedades aptitud, velocidad, edad, equilibrio mental y casta de origen son pertinentes para dicha actividad y se tienen en cuenta. Si nos interesamos por los caballos en su actividad de bestia de tiro, las propiedades edad, tamaño, fuerza y corpulencia son pertinentes para dicha actividad y se tienen en cuenta.

La abstracción es una simplificación indispensable para el proceso de modelado. Un objeto UML es una abstracción de un objeto del mundo real de acuerdo con las necesidades del sistema de la cual sólo se tienen en cuenta los elementos esenciales.

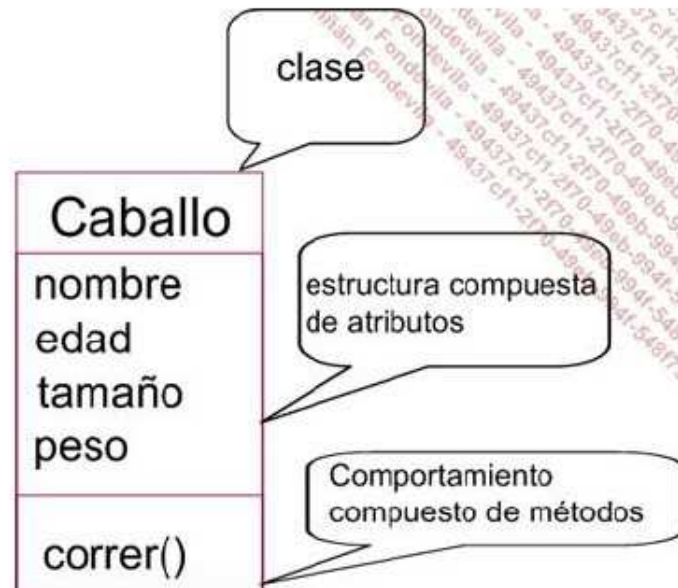
2.3. Clases de objetos

Un conjunto de objetos similares, es decir, con la misma estructura y comportamiento, y constituidos por los mismos atributos y métodos, forma una clase de objetos. La estructura y el comportamiento pueden entonces definirse en común en el ámbito de la clase.

Todos los objetos de una clase, llamada también instancia de clase, se distinguen por tener una identidad propia y sus atributos les confieren valores específicos.

Ejemplo

El conjunto de caballos constituye la clase Caballo, que posee la estructura y el comportamiento descritos en la siguiente figura:



La clase Caballo

El caballo Jorgelina es una instancia de la clase Caballo cuyos atributos y valores se ilustran en la siguiente figura:



El caballo Jorgelina

El nombre de las clases se escribe en singular y está formado siempre por un nombre precedido o seguido por uno o varios adjetivos que lo califican. Dicho nombre es revelador de los objetos que forman la clase.

2.4. Encapsulación

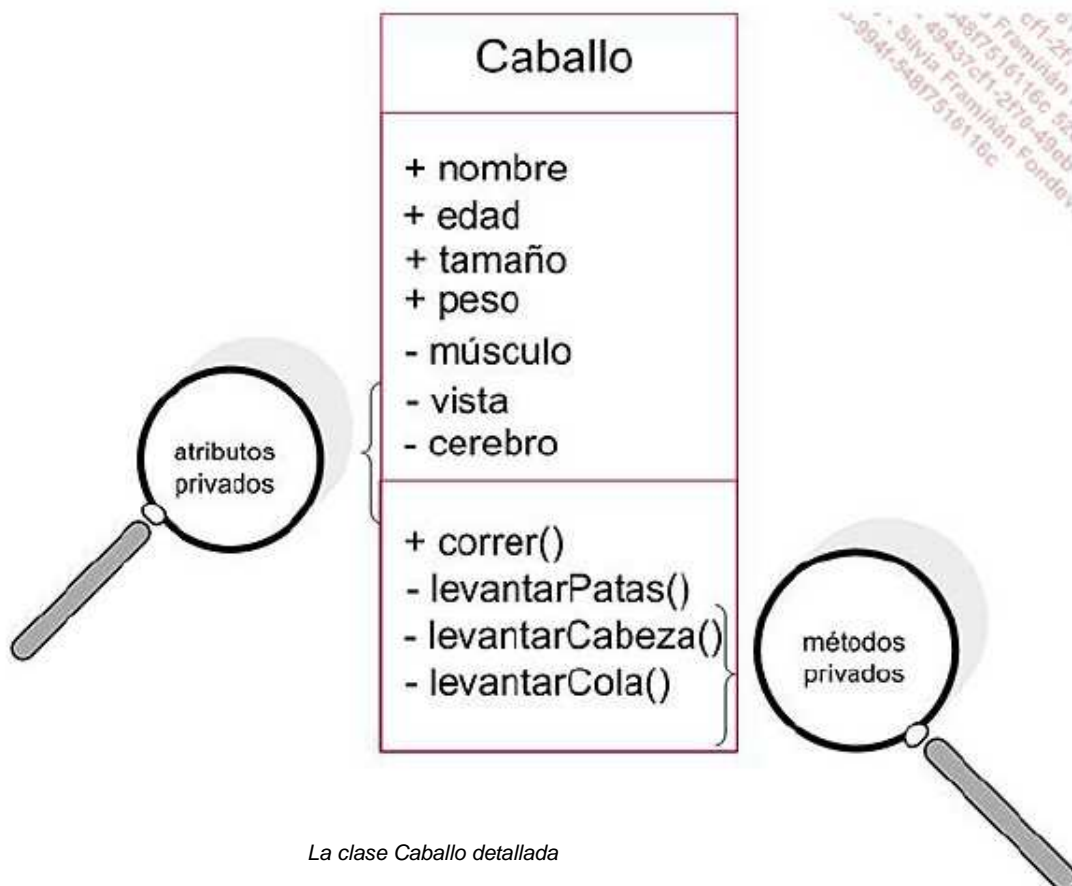
La encapsulación consiste en ocultar los atributos y métodos del objeto a otros objetos. En efecto, algunos atributos y métodos tienen como único objetivo tratamientos internos del objeto y no deben estar expuestos a los objetos exteriores. Una vez encapsulados, pasan a denominarse atributos y métodos privados del objeto.

La encapsulación es una abstracción, ya que se simplifica la representación del objeto con relación a los objetos externos. Esta representación simplificada está formada por atributos y métodos públicos del objeto.


La definición de encapsulación se realiza en el ámbito de la clase. Los objetos externos a un objeto son, por tanto, las instancias de las demás clases.

Ejemplo

Al correr, los caballos efectúan diferentes movimientos como pueden ser levantar las patas, levantar la cabeza o levantar la cola. Esos movimientos son internos al funcionamiento del animal y no tienen por qué ser conocidos en el exterior. Son métodos privados. Las operaciones acceden a una parte interna del caballo: sus músculos, su cerebro y su vista. La parte interna se representa en forma de atributos privados. El conjunto de atributos y métodos se ilustra en la siguiente figura.



La clase Caballo detallada

 En la notación UML, los atributos y métodos públicos aparecen precedidos del signo más mientras que los privados (encapsulados) aparecen precedidos del signo menos.

2.5. Especialización y generalización

Hasta el momento, cada clase de objetos se introduce de forma separada a las demás clases. Las clases pueden definirse también como subconjuntos de otras clases, pero éstos deben constituir siempre conjuntos de objetos similares.

Hablamos entonces de subclases de otras clases que, por tanto, constituyen especializaciones de esas otras clases.

Ejemplo

La clase de los caballos es una subclase de la clase de los mamíferos.

La generalización es la relación inversa a la especialización. Si una clase es una especialización de otra clase, ésta última es una generalización de la primera. Es su superclase.

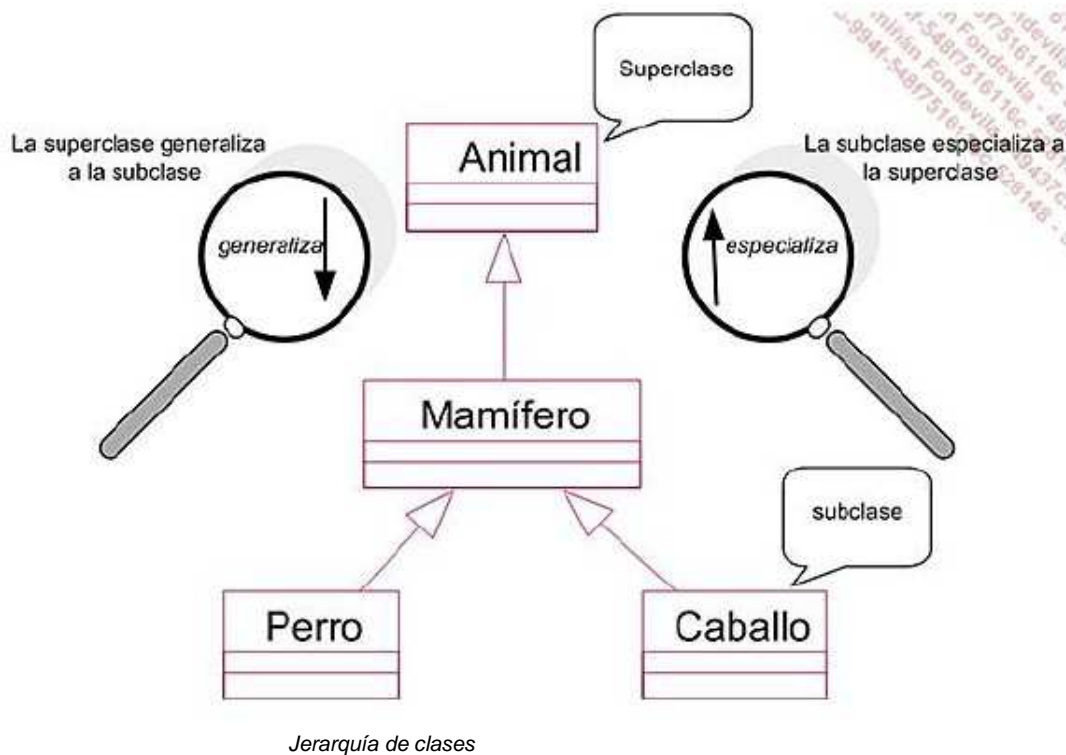
Ejemplo

La clase de los mamíferos es una superclase de la clase de los caballos.

La relación de especialización puede aplicarse a varios niveles, dando lugar a la jerarquía de clases.

Ejemplo

La clase de los caballos es una subclase de la clase de los mamíferos, ella misma subclase de la clase de los animales. La clase de los perros es otra subclase de la clase de los mamíferos. La jerarquía de clases correspondiente aparece representada a continuación.

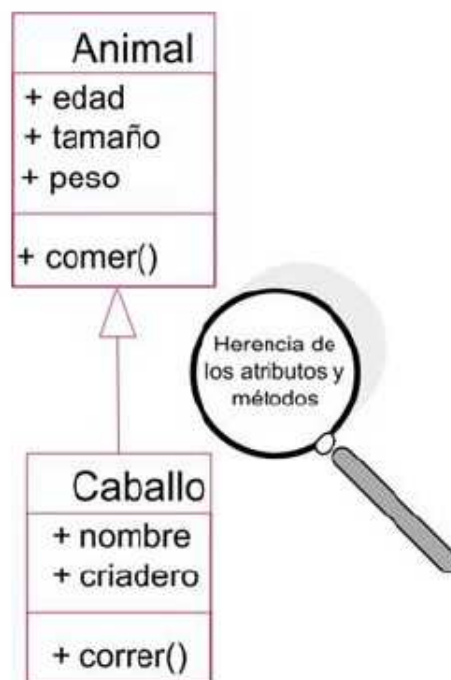


2.6. Herencia

La herencia es la propiedad que hace que una subclase se beneficie de la estructura y del comportamiento de su superclase. La herencia deriva del hecho de que las subclases son subconjuntos de las superclases. Sus instancias son asimismo instancias de la superclase y, por consiguiente, además de la estructura y del comportamiento introducido en la subclase, se benefician también de la estructura y comportamiento definidos por la superclase.

Ejemplo

Tomamos un sistema en el que la clase Caballo es una subclase directa de la clase Animal. El caballo se describe entonces mediante la combinación de la estructura y del comportamiento derivados de las clases Caballo y Animal, es decir, mediante los atributos edad, tamaño, peso, nombre y casta así como los métodos comer y correr.



Herencia

La herencia es una consecuencia de la especialización. Sin embargo, los informáticos emplean mucho más a menudo el término hereda que especializa para designar la relación entre una subclase y su superclase.

2.7. Clases abstractas y concretas

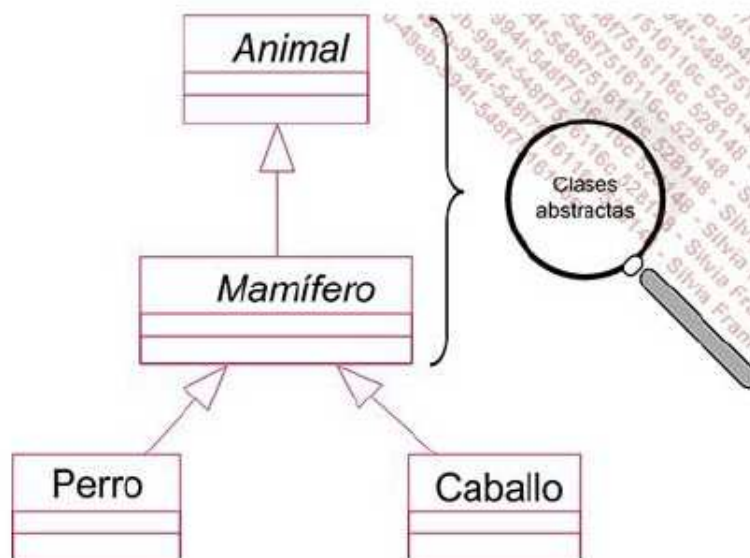
Si examinamos la jerarquía presentada en la figura vemos que en ella existen dos tipos de clases:

- Las clases que poseen instancias, es decir, las clases Caballo y Perro, llamadas clases concretas .
- Las clases que no poseen directamente instancias, como la clase Animal. En efecto, si bien en el mundo real existen caballos, perros, etc., el concepto de animal propiamente dicho continúa siendo abstracto. No basta para definir completamente un animal. La clase Animal se llama clase abstracta .

La finalidad de las clases abstractas es poseer subclases concretas, éstas sirven para factorizar atributos y métodos comunes a las subclases.

Ejemplo

La figura retoma la jerarquía detallando las clases abstractas y las clases concretas. En UML, el nombre de las clases abstractas se escribe en cursiva.



Clases abstractas y concretas

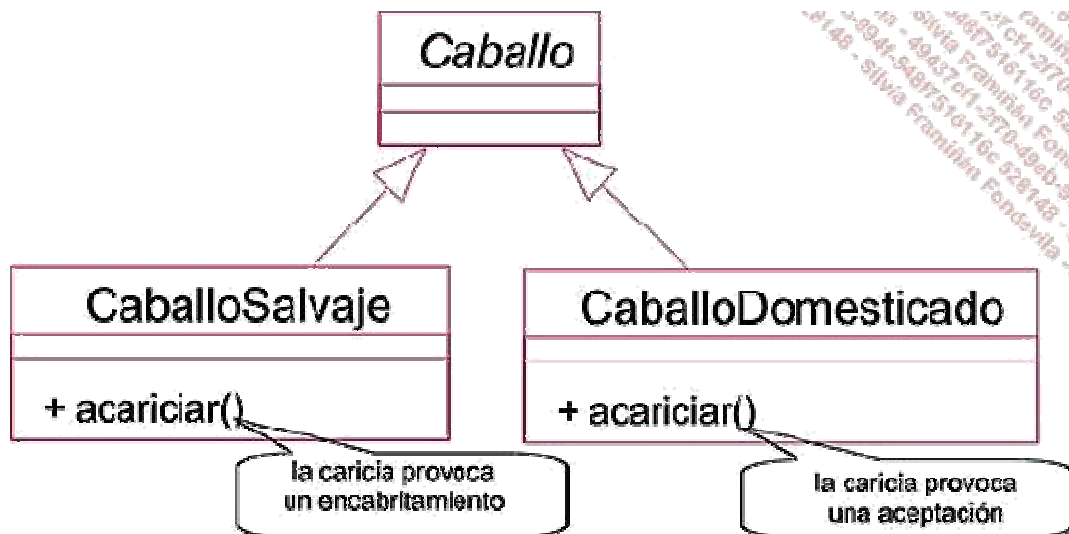
2.8. Polimorfismo

El polimorfismo significa que una clase (generalmente abstracta) representa un conjunto formado por objetos diferentes, ya que éstos son instancias de subclases diferentes. Cuando se llama a un método del mismo nombre, esta diferencia se traduce en comportamientos distintos (excepto en los casos en los que el método es común y las subclases lo han heredado de la superclase).

Ejemplo

Tomemos la jerarquía de clases ilustrada en la siguiente figura. El método `acariciar` tiene un comportamiento diferente según si el caballo es una instancia de `CaballoSalvaje` o de `CaballoDomesticado`. En el primer caso, el comportamiento será un rechazo (que se traducirá en un encabritamiento), mientras que en el segundo el comportamiento será una aceptación.

Si consideramos la clase `Caballo` en su totalidad, tenemos un conjunto de caballos que reaccionan de distinta manera al activarse el método `acariciar`.



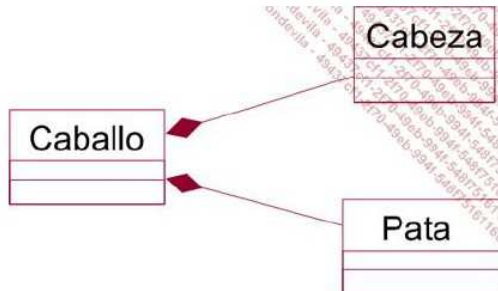
Polimorfismo

2.9. Composición

Un objeto puede ser complejo y estar compuesto por otros objetos. La asociación que une a estos objetos es la composición, que se define a nivel de sus clases, pero cuyos vínculos se establecen entre las instancias de las clases. Los objetos que forman el objeto compuesto se denominan componentes.

Ejemplo

Un caballo es un ejemplo de objeto complejo. Está formado por diferentes órganos (patas, cabeza, etc.). La representación gráfica de esta composición puede verse en la siguiente figura.



Composición

La composición puede adoptar dos formas:

- composición débil o agregación ;
- composición fuerte.

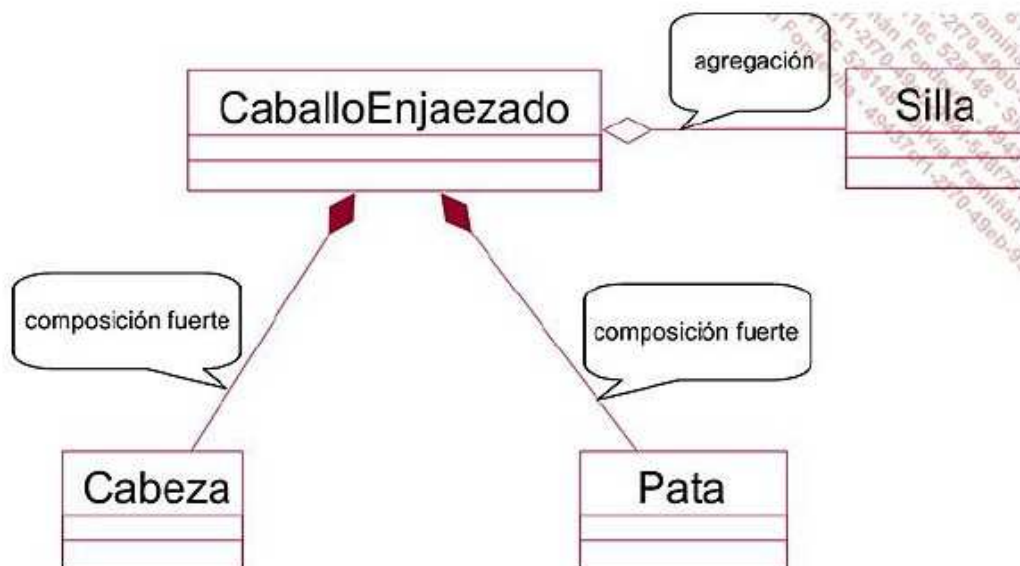
En la composición débil, los componentes pueden ser compartidos por varios objetos complejos. En la composición fuerte, los componentes no pueden compartirse y la destrucción del objeto compuesto conlleva la destrucción de sus componentes.

Ejemplo

Si retomamos el ejemplo precedente con el supuesto de un caballo de carreras enjaezado y añadimos a sus componentes una silla, obtenemos:

Una composición fuerte para las patas y la cabeza. Efectivamente, las patas y la cabeza no pueden compartirse y la desaparición del caballo conlleva la desaparición de sus órganos.

Una agregación o composición débil para la silla.



2.10. La especialización de los elementos: la noción de estereotipo en UML

Ahora introduciremos los estereotipos de UML cuya finalidad es especializar dichos conceptos.

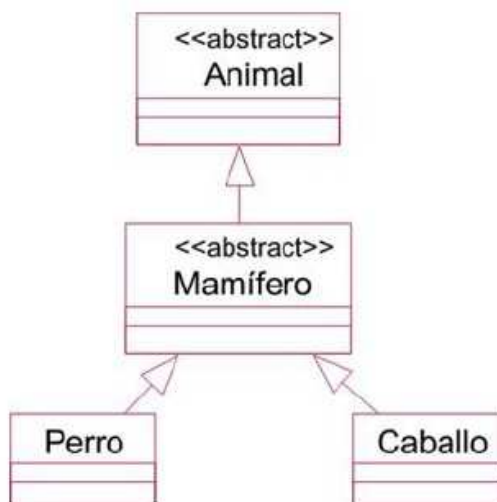
Los estereotipos están formados por palabras clave que explicitan la especialización. Estas palabras clave aparecen entre comillas.

La especialización se realiza independientemente del sistema que se desee modelar.

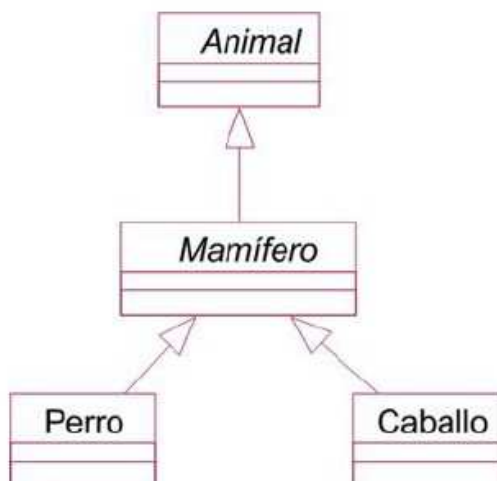
Ejemplo

El concepto de clase abstracta es un concepto especializado del concepto de clase. Hemos visto que una clase abstracta se representa como una clase con un nombre en cursiva. Esta representación gráfica incluye un estereotipo implícito, pero también podemos prescindir de poner el nombre de la clase en cursiva y precisar de forma explícita el estereotipo <<abstract>>.

Presentamos el estereotipo explícito en la siguiente figura. Éste puede emplearse al escribir a mano diagramas UML.



Estereotipo explícito << abstract>>



Estereotipo implícito << abstract>>