Universidad Nacional Abierta y a Distancia

MODELADO DE DATOS

Diagrama de Clases

Descripción

Concepto de diagrama de clases, características y componentes..

Luis Manuel Márquez Rodríguez
Ingeniería de Sistemas

Tabla de Contenido

Diagrama de Clases

Son la parte más importante del diseño, permite observar las relaciones, atributos y métodos (operaciones) de todo el sistema que se creará. Se hará referencia a los componentes del mismo y cómo se grafican las relaciones entre las diversas clases.

- **Objeto:** es una abstracción o cosa con límites bien definidos y con significado para el problema que se está manejando. Cada objeto presenta una identidad que lo distingue de otros, también un estado y un comportamiento. (Mediavilla, s.f.)

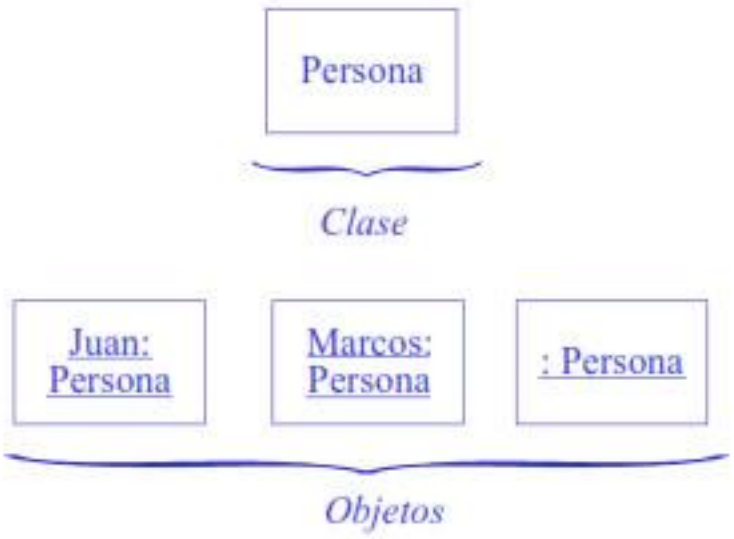


Figura 1. Clases y objetos

Fuente: Mediavilla, E. (s.f.). Programación orientada a Objetos. Master de Computación. II Modelos y Herramientas UML, Modelado estructural. Recuperado de http://www.ctr.unican.es/asignaturas/MC_OO/Doc/M_Estructural.pdf

Todos los objetos son instancias de una clase y la clase de un objeto es una propiedad implícita del objeto.

- **Clase:** comprende un conjunto de objetos con propiedades similares (atributos), con un comportamiento (métodos), que se relacionan con otros objetos.

Las clases se representan con un rectángulo con 3 divisiones, en la primera se coloca el nombre de la clase, en el segundo se ubican los atributos y en el último aparecerán los métodos. Las clases se dividen en clases padres y clases hijas, la primera es la principal y las segundas son las que son, por decirlo así, una copia de la principal, es decir que pueden heredar sus atributos y métodos.

Al hablar de herencia se hace relación a "una propiedad esencial de la Programación Orientada a Objetos que consiste en la creación de nuevas clases a partir de otras ya existentes", esta permite que el código pueda ser reutilizado. (Franco, 2001)

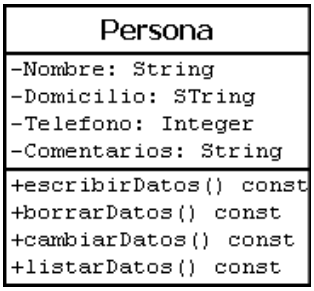


Figura 2. Ejemplo de clase

- **Atributos:** representan las propiedades de la clase, cada nombre de atributo es único dentro de una clase, pero cada atributo tiene un valor para cada instancia de la clase. Un atributo es un valor, no un objeto.

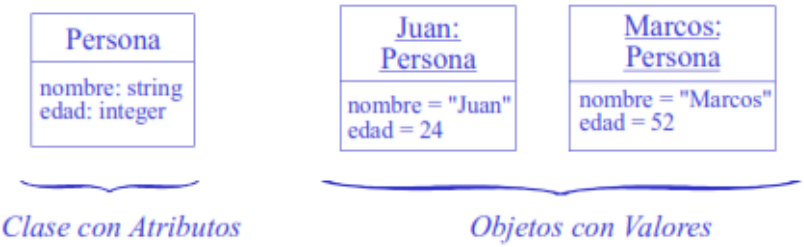


Figura 3. Ejemplo de atributos de la clase y objetos con valores

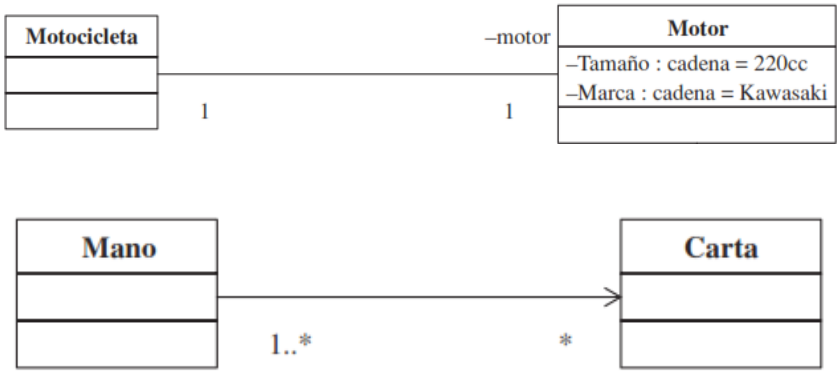
Fuente: Mediavilla, E. (s.f.). Programación orientada a Objetos. Master de Computación. II Modelos y Herramientas UML, Modelado estructural. Recuperado de http://www.ctr.unican.es/asignaturas/MC_OO/Doc/M_Estructural.pdf

- **Métodos:** son las acciones u operaciones de la clase, que permiten que esta interactúa con su entorno. Como se puede observar en la figura 1, a los métodos (parte última del rectángulo), se les adiciona un símbolo, que de acuerdo a este indica si se trata de un método público ("+", es decir que se pueden consultar externamente); privado ("-", solo puede accederse de forma interna) y protegido ("#", solo es accesible por clases hijas).

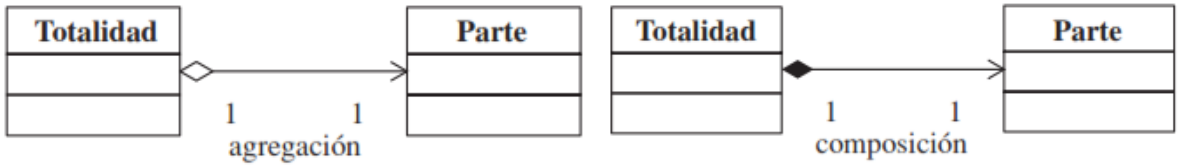
- **Relaciones:** para relacionar una clase con otra se establecieron unos conectores que dependiendo de su contextura determinarán qué tipo de relación establecen (herencia, generalización, realización, composición, agregación, dependencia y asociación).

Como se explicó en el tema anterior (lectura 2), la cardinalidad de las relaciones. en UML, indican el grado y nivel de dependencia, se escriben en cada extremo de la relación, estas son: Uno o muchos: 1..* (1..n); 0 o muchos: 0..* (0..n) y número fijo: m (m denota el número).

Tabla 1. Tipo de relaciones



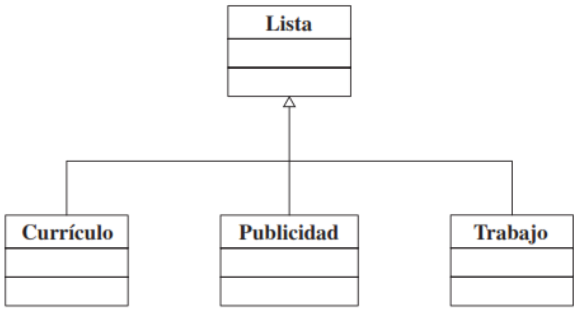
El conector de asociación es una línea continua. Si no hay flechas es bidireccional. Si por el contrario, aparece en cualquier extremo de esta, se dice que la asociación es dirigida o direccional.
El tiempo de vida de un objeto no depende del tiempo de vida del incluido. La relación de asociación permite asociar objetos que colaboran entre sí.



El conector de agregación se representa con una línea continua, en el extremo que sale del clasificador de totalidad se dibuja un diamante y a otro clasificador de parte, se dibuja una flecha. El conector de composición, difiere únicamente en que el diamante está relleno.

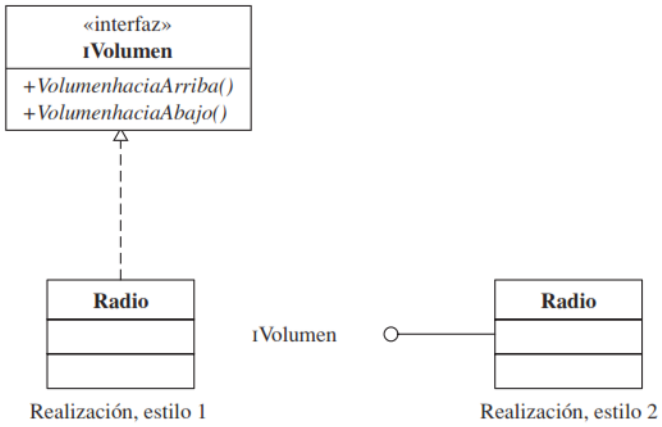
Para modelar objetos complejos se deben tener en cuenta los tipos de datos básicos (enteros, reales y secuencia de caracteres); si son objetos instanciados de clases definidas, se puede hacer relación a ellos de forma estática (por valor), es decir que el tiempo de vida del objeto está condicionado al tiempo de vida del que lo incluye (relación de composición).

La relación de agregación es dinámica, en donde el tiempo de vida del objeto es independiente del que lo incluye (por referencia), el objeto base utiliza al incluido para su funcionamiento.



El conector de herencia es una flecha hueca, con línea continua que parte de la clase hija hacia la clase padre o base.

Indica que una subclase hereda los métodos y atributos especificados por una Súper Clase, por ende la Subclase además de poseer sus propios métodos y atributos, poseerá las características y atributos visibles de la Súper Clase (*public* y *protected*). (Salinas, 1996)



El conector de realización es una línea punteada con un triángulo hueco. Las relaciones de realización se refieren a la herencia de interfaces de realización o las propias interfaces.

El conector de dependencia o instanciación (uso) se dibuja con una línea punteada y con una flecha que va de la clase dependiente (o clase cliente) hacia la clase independiente (o clase proveedor).

La instanciación de la clase es dependiente de otro objeto/clase.

Casos Particulares

Clase Abstracta: es aquella que no puede ser instanciada porque sus métodos son abstractos, es decir, aún no han sido implementados. Para utilizarla es necesario crear subclases que implementen los métodos abstractos definidos.

Su representación se realiza con el nombre de la clase y sus métodos en letra cursiva o itálica.

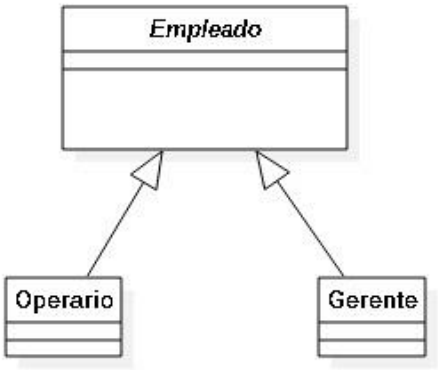


Figura 4. Representación de una clase abstracta

Clase Parametrizada: en su representación gráfica se coloca un subcuadro en el extremo superior de la clase, allí se escriben los parámetros que deben ser pasados a la clase para que esta pueda ser instanciada.

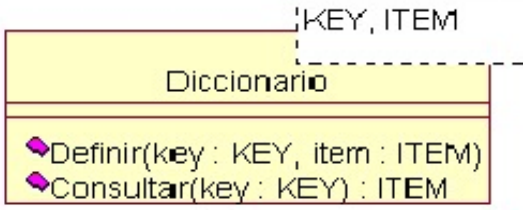


Figura 5. Representación de una clase parametrizada

Fuente: Salinas, P. (1996). UML: Modelo de clases. Departamento de Ciencias de la Computación. Universidad de Chile. Recuperado de https://users.dcc.uchile.cl/~psalinas/uml/modelo.html

Notas

Se representan usando llaves {}, indican las observaciones, condiciones semánticas o restricciones.

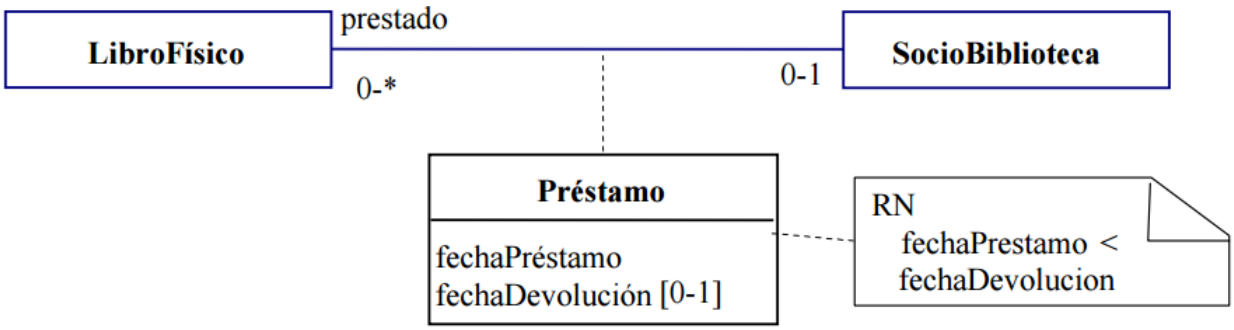


Figura 6. Ejemplo del uso de notas

Se recomienda imponer restricciones, especificándolas con el Lenguaje de Restricciones de Objeto (OCL, por sus siglas en inglés). Las restricciones permiten comprobar "las características del perfil y la consistencia entre los diagramas del modelo, a saber, diagrama de clases y diagrama de secuencia". (Universidad de Mendoza, s.f.).

Referencias Bibliográficas

Franco, A. (2000). Curso de Lenguaje Java: La clase base y la clase derivada. Recuperado de http://www.sc.ehu.es/sbweb/fisica/cursoJava/fundamentos/herencia/herencia.htm

Kimmel, P. (2006). Manual de UML. México: McGraw-Hill, p. 1-12.

Mediavilla, E. (s.f.). Programación orientada a Objetos. Master de Computación. II Modelos y Herramientas UML, Modelado estructural. Recuperado de http://www.ctr.unican.es/asignaturas/MC_OO/Doc/M_Estructural.pdf

Salinas, P. (1996). UML: Modelo de clases. Departamento de Ciencias de la Computación. Universidad de Chile. Recuperado de https://users.dcc.uchile.cl/~psalinas/uml/modelo.html

Universidad de Mendoza (s.f.). Validación de Patrones de Diseño de Comportamiento a través de Perfiles UML. Tecnología. Mendoza, Argentina, p. 24. Recuperado de http://www.um.edu.ar/es/contenido/institutos/ingenieria/tecnologia/ValidacionPatrones.pdf