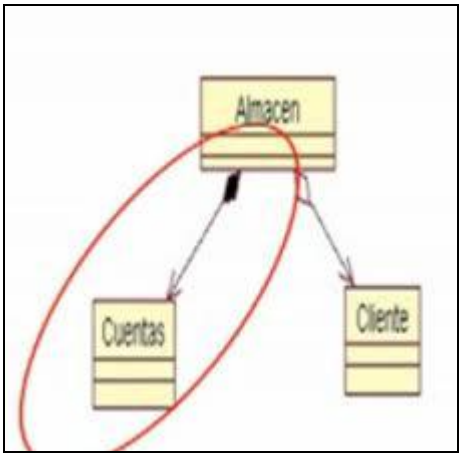
Tecnicatura
Superior en Programación

UNIDAD II

RELACIONES DE COMPOSICIÓN AGREGACIÓN



Temario

1. Tipos de Relaciones entre clases
    1.1 Relaciones de Asociación
    1.2 Relaciones de Clientela
        1.2.1 Agregación
        1.2.2 Composición
    1.3 Relaciones de Generalización/Especialización
2. Ejercitación Práctica propuesta

Unidad II: Arreglos

1. Tipos de Relaciones entre Clases

Durante la ejecución de un programa, de las clases planteadas surgen los diversos objetos que lo componen y han de interactuar entre sí para lograr una serie de objetivos comunes.
Existen varios tipos de relaciones que pueden unir a los diferentes objetos, pero entre ellas destacan las relaciones de: asociación, agregación / composición, y generalización / especialización.

1.1 Relaciones de Asociación

Serían relaciones generales, en las que un objeto realiza llamadas a los servicios (métodos) de otro, interactuando de esta forma con él. Representan las relaciones con menos riqueza semántica. Este tipo de relaciones entre objetos las hemos graficado en la ejercitación que hemos desarrollado en la semana 1 y 2 de clases. Por ej: El objeto aplicación le solicita al objeto alumnos que muestre su nombre por medio del mensaje getLegajo().

1.2 Relaciones de Clientela: Agregación/Composición

Muchas veces una determinada entidad existe como conjunción de otras entidades, como un conglomerado de ellas. La orientación al objeto recoge este tipo de relaciones como dos conceptos; **la agregación** y **la composición**.

1.2.1 Agregación

La agregación es un ***tipo de asociación que indica que una clase es parte de otra clase (composición débil)***. Los componentes pueden ser compartidos por varios compuestos (de la misma asociación de agregación o de varias asociaciones de agregación distintas). La destrucción del compuesto no conlleva la destrucción de los componentes. Habitualmente se da con mayor frecuencia que la composición.

La agregación se representa en UML mediante un diamante de color blanco colocado en el extremo en el que está la clase que representa el "todo".

Veamos un ejemplo de agregación:

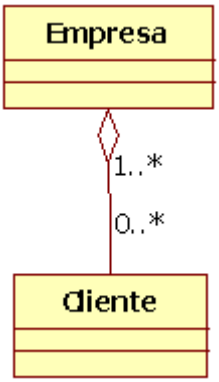


Figura 1: Ejemplo de Agregación

- Tenemos una clase Empresa.
- Tenemos una clase Cliente.
- Una empresa agrupa a varios clientes.

1.2.2 Composición

Composición ***es una forma fuerte de composición*** donde la vida de la clase contenida debe coincidir con la vida de la clase contenedor. Los componentes constituyen una parte del objeto compuesto. De esta forma, los componentes no pueden ser compartidos por varios objetos compuestos. La supresión del objeto compuesto conlleva la supresión de los componentes.

El símbolo de composición es un diamante de color negro colocado en el extremo en el que está la clase que representa el "todo" (Compuesto).
Veamos un ejemplo de composición:

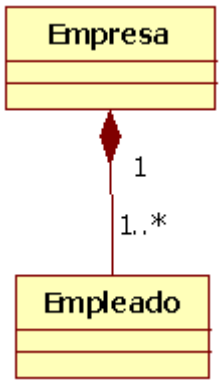


Figura 2: Ejemplo de Composición

Las principales diferencias entre la composición y agregación, se detalla en la siguiente tabla:

	Agregación	**Composición**
Varias asociaciones comparten los componentes.	**SI**	**NO**
Destrucción de los componentes al destruir el compuesto.	**NO**	**SI**

Tabla 1: Diferencia entre Agregación y Composición

En resumen: En líneas generales, como hemos visto, se podría decir que **la diferencia entre agregación y composición es conceptual**, no se diferencia por código, o al menos, en el mayor de los casos y en la mayoría de los lenguajes de programación como Java.

Relaciones de Generalización/Especialización (Herencia)

A veces sucede que dos clases tiene muchas de sus partes en común, lo que normalmente se abstrae en la creación de una tercera clase (padre de las dos) que reúne todas sus características comunes.

El ejemplo más extendido de este tipo de relaciones es la herencia, propiedad por la que una clase (clase hija) recoge aquellos métodos y atributos que una segunda clase (clase padre) ha especificado como "heredables".

Este tipo de relaciones es característico de la programación orientada a objetos.

En realidad, la generalización y la especialización son diferentes perspectivas del mismo concepto, la generalización es una perspectiva ascendente (bottom-up), mientras que la especialización es una perspectiva descendente (top-down). Este tipo de relación entre clases lo veremos más adelante.

***2.* Ejercitación Práctica Propuesta**

a. Una Veterinaria encargada del cuidado de mascotas ha solicitado la realización de un software para obtener resultados sobre sus clientes y sus respectivas mascotas. (Suponer que cada cliente posee una sola mascota)
Se sabe que cada **Cliente** de esta veterinaria tienen: un numero de cliente, un nombre, una antigüedad (hace cuanto que son clientes de la veterinaria) y una Mascota.
De la **Mascota** se dispone los siguientes datos: el nombre y la edad.
El software requiere cargar un número n de Clientes (este valor deberá ser cargado por teclado)
Almacenar la información de los clientes en un arreglo.

Se pide:

- Mostrar la cantidad de clientes.
- Mostrar como resultado el promedio de edad de las mascotas.
- Informar cuantos clientes tienen una antigüedad mayor igual a 5 años.

b. Una empresa necesita desarrollar un programa que permite obtener información de los comprobantes (recibos) que emite la empresa.
Se han detectado en la etapa de relevamiento que cada **Recibo** tiene los siguientes datos: nro de recibo, fecha, importe y un cliente.
Cada **Cliente** tiene un código de identificación, apellido y nombre, tipo de cliente (1-Regular o 2-No Regular).
Almacenar los recibos en un arreglo.
Se pide:

- Mostrar el total de importe de los recibos, correspondiente a los clientes que son regulares.
- Mostrar el nombre y apellido del cliente que tiene un importe menor en el recibo.
- Calcular el promedio de importe de los recibos, correspondiente a los clientes no regulares.
- Informar la cantidad de clientes regulares y no regulares.

c. Un local que se dedica a la reparación de PC, nos ha solicitado la implementación de un software que facilite la registración de los mantenimientos realizados a los clientes.
Cada **Mantenimiento** tiene los siguientes datos: nro de mantenimiento (es atributo estático), un técnico, un cliente, una PC (suponer que un cliente trae sólo una PC), diagnóstico, importe del mantenimiento.
Un **Técnico** tiene la siguiente información: nro de técnico (puede ser 1,2 o 3), apellido y nombre.
Un **Cliente** tiene los siguientes datos: número de cliente, apellido y nombre, domicilio.

Y cada **PC** tiene: marca, modelo, tamaño disco, memoria.
Almacenar los mantenimientos en un arreglo.

Se pide:

- Informar el importe de los diagnósticos efectuados por el técnico número 3.
- Mostrar los datos de los clientes que han efectuado un mantenimiento, cuyo monto es superior a un valor "x" ingresado por teclado.
- Mostrar los datos de la PC y del cliente, de un determinado cuyo nro de mantenimiento debe ser ingresado por teclado.