Unidad

3

DIPLOMATURA EN PROGRAMACION JAVA

Capítulo 3

Relaciones

Relaciones

Asociaciones y enlaces

Tanto las clases como los objetos se relacionan de diferentes maneras. Cada relación depende de cómo interaccionan unos con otros. Por ejemplo, se puede declarar un objeto como atributo de una clase, pasar un objeto como parámetro a un método, devolver un objeto desde un método o declararlo como un elemento de un vector de objetos. Todas estas posibilidades dependen siempre de la capacidad del lenguaje utilizado, pero más allá de la cuestión técnica, definen una relación en la cual se establece como interactúan dos objetos del tipo de las clases a las que pertenecen los miembros de dicha relación.

Por eso, se puede definir a toda relación entre clases y objetos como una asociación que puede ser diagramada en UML. Para indicar que dos clases u objetos están relacionados por medio de una asociación, se diagrama una línea que une a ambos a la cual se denomina enlace. Resumiendo

Asociación:

- Se refiere a una relación representada por una línea en un diagrama de colaboración de clases u objetos
- La línea puede ser horizontal o vertical
- La línea de la relación posee un nombre que describe la relación

Enlace:

- Se refiere a la relación mostrada en un diagrama de clases u objetos entre dos de ellos

Asociaciones

La representación de las asociaciones y sus enlaces en un diagrama de objetos es como se presenta a continuación

En el caso de los objetos, el diagrama puede ser calificado o no. Esto quiere decir, si se conoce el nombre del objeto se lo coloca antecediendo al nombre de la clase que define su tipo, pero si no se posee dicha información al momento de realizar el diagrama, este se puede hacer poniendo sólo el nombre de la clase antecediéndolo con dos puntos y subrayándolo, como muestra el diagrama anterior. El siguiente ejemplo muestra cuando se determino inclusive el nombre de los objetos

El nombre que se ve en el enlace es una frase verbal que define el tipo de asociación que se refleja en el enlace. Así, por ejemplo, este último diagrama puede leerse:

Juan, que es del tipo Persona, posee un Fiat que es del tipo Automóvil

Relaciones entre clases

Los diagramas de clases muestran los elementos definidos dentro de ellas total o parcialmente, pero no indican como interaccionan con otras, ya sea desde el punto de vista del diseño como en tiempo de ejecución.

Se pueden diagramar las interacciones entre clases desde dos puntos de vista, el estático y el dinámico. El punto de vista estático esta asociado con la descripción del tipo de interacción que se define entre dos clases mientras que, el dinámico, muestra el comportamiento de los objetos del tipo de dichas clases cuando se utiliza esa relación en tiempo de ejecución.

A los diagramas estáticos se los denomina diagramas de clases, mientras que a los dinámicos se los denomina diagramas de objetos.

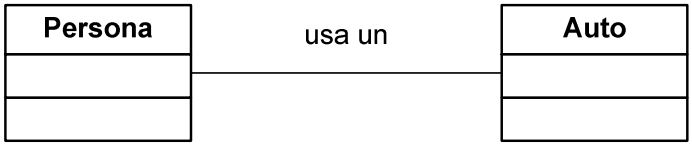
Las relaciones entre clases definen una asociación y se las puede clasificar según el tipo de interacción que definen. Los tipos que definen la mayoría de los lenguajes orientados a objetos son:

- Asociaciones simples
- Agregaciones
- Composiciones
- Herencia

Asociaciones simples

El primer paso para representar la relación que existe entre dos clases, es dibujar una línea entre ellas con el nombre que se le asigna a esta. La sola existencia de una línea entre dos clases define una asociación. Dentro de las posibles asociaciones existen las más débiles llamadas asociaciones simples

Para detectar una asociación simple entre dos clases u objetos se utilizan las palabras "**usa un**" o "**usa una**". Por ejemplo, "una persona usa un auto". Partiendo de este ejemplo, se puede realizar un diagrama como el que muestra la figura:



Este tipo de asociaciones muestran la interacción entre dos clases y definen su comportamiento. Es claro que la relación entre la persona y el auto es temporal e independiente de la vida de los objetos involucrados. Por ejemplo, el hecho que la "persona use el auto" no implica que la persona cree el auto para usarlo.

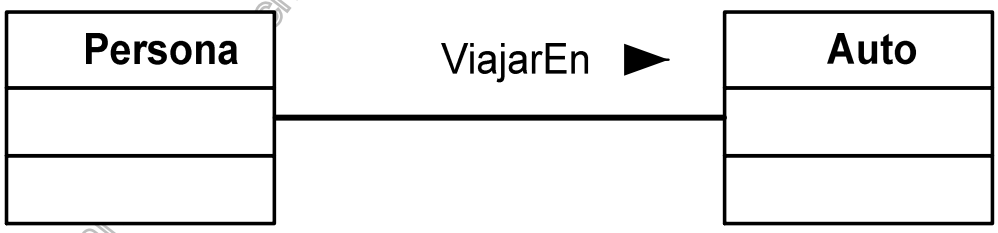
La asociación es inherentemente bidireccional, lo cual significa que la puede crear de cualquiera de las clases que intervienen para definir un recorrido "lógico" en ambas direcciones.

Nota: El recorrido es puramente abstracto, **no** se trata de una sentencia sobre conexiones entre entidades de software.

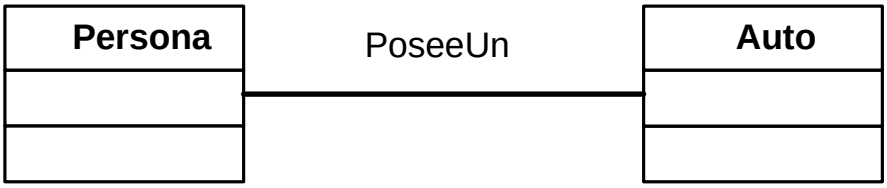
Una "flecha de dirección de lectura" opcional indica la dirección de cómo leer el nombre de la asociación. NO indica ningún tipo de visibilidad o navegación. Si no esta presente, por convención se lee de izquierda a derecha o de arriba hacia abajo.

Nota: la flecha de dirección de lectura no tiene significado en términos del modelo, sólo es una ayuda para quien lee el diagrama.

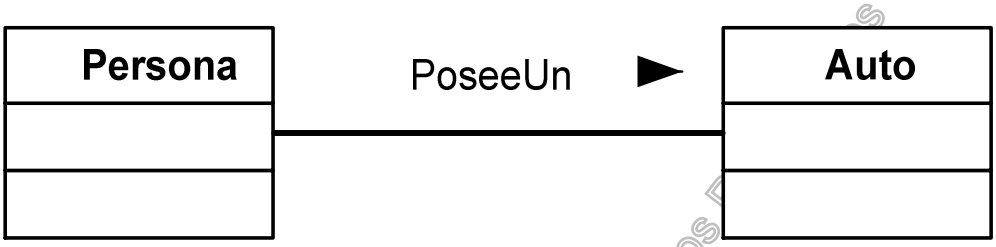
Por otro lado, se le puede asignar a una relación un nombre y una dirección. Por ejemplo, si para definir la interacción entre dos clases se enuncia que: una persona usa un auto para viajar, el diagrama descriptivo podría ser:



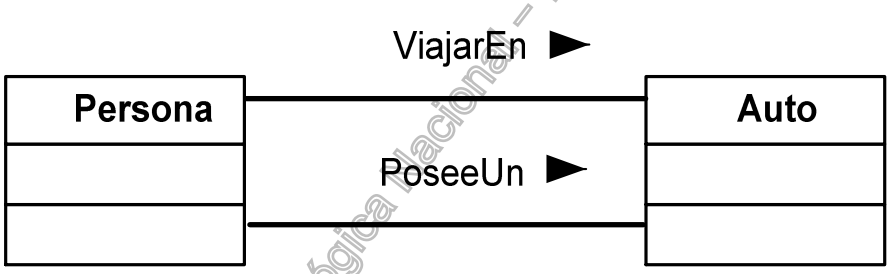
En este caso, la asociación simple se define como "viajarEn". La dirección de la asociación indica la forma en que se relacionan estas clases, NO la navegabilidad.



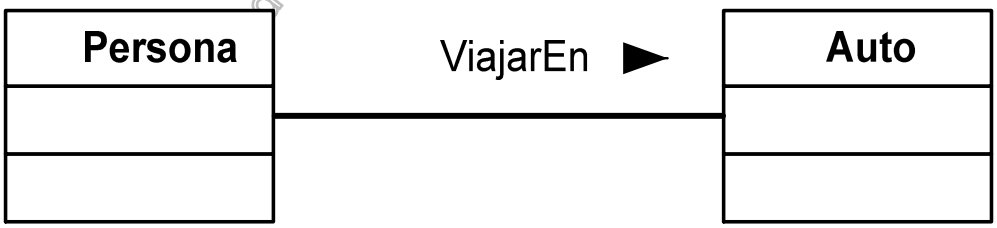
Dos clases pueden tener más de una asociación, por ejemplo para las mismas clases si se quiere representar que la persona posee un auto (siguiendo la convención), el diagrama seria:



El cual deberá leerse según la convención como si el diagrama fuera:



Como en el caso anterior se puede observar dos asociaciones entre las mismas clases, ambas se pueden colocar en un mismo gráfico, como muestra la figura:



Asignación de nombres en las asociaciones

Para asignar un nombre a una asociación, se debe seguir la siguiente guía:

NombreTipo ***FraseVerbal*** NombreTipo

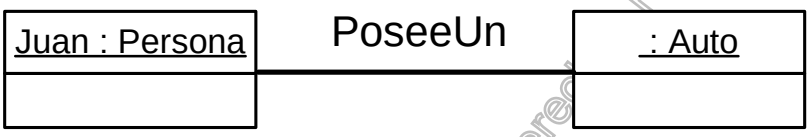
Donde frase verbal es un nombre que califica la relación, en este caso los nombres utilizados son:

- ViajarEn
- PoseeUn

Otro formato válido es

- Viajar-en
- Posee-un

Cuando se diagrama una asociación en tiempo de ejecución (diagrama dinámico) se diferencia del diagrama estático porque el nombre del objeto antecede al de la clase y se los separa con dos puntos. Ambos nombres se colocan subrayados. Si no se conociera el nombre del objeto cuando se realiza el diagrama, sólo se colocan los dos puntos y el nombre de la clase. Ambas situaciones se muestran en el diagrama:



O también:

Roles

Cada extremo de una asociación es un rol. Los nombres de rol proporcionan la herramienta para identificar cada extremo de una relación y son los que sirven para navegarla en caso de requerirlo. El nombre deberá ser único en ambos lados para diferenciar cada extremo de la asociación y por lo general es el mismo nombre de la variable de referencia que se utiliza en un lenguaje de programación para declarar la relación.

Si en el diagrama existe más de una asociación, todos los nombres de roles en cada una de las que se muestren deberán ser diferentes para no crear ambigüedades.

A pesar de todas estas restricciones, los nombres de roles son opcionales y pueden ser omitidos.

Los roles pueden tener, opcionalmente:

- Nombre
- Expresión de multiplicidad
- Navegabilidad

- Visibilidad

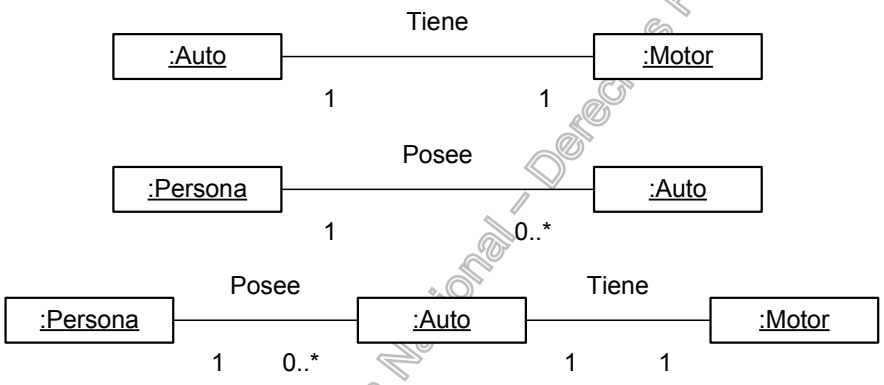
Pueden especificarse las cuatro opciones, ninguna, o cualquier combinación de ellas.

Asociaciones y Multiplicidad

La multiplicidad define cuantas instancias de una clase, por ejemplo, A pueden asociarse con una instancia de otra clase B.

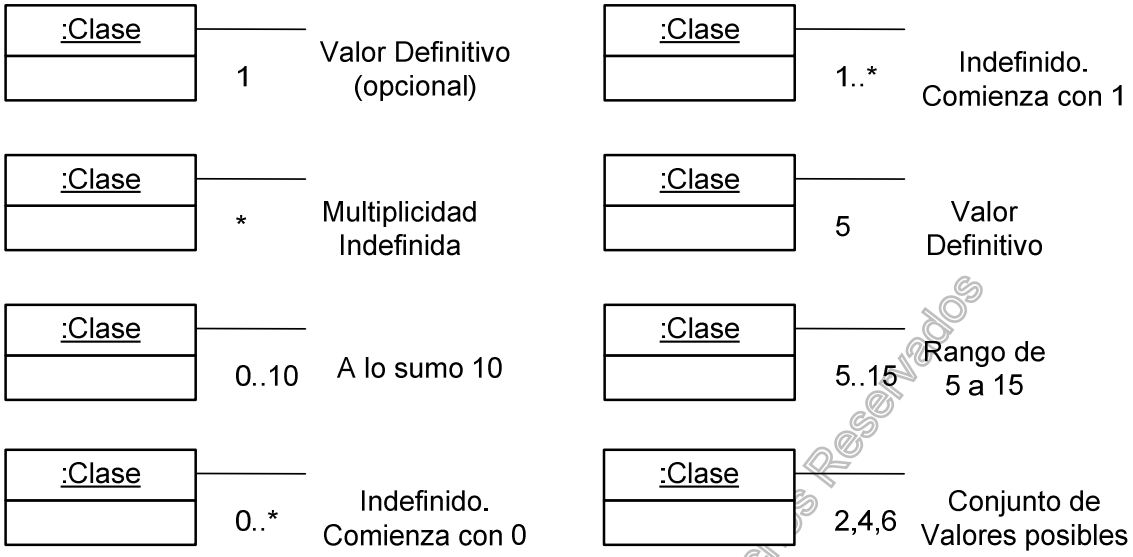
Las multiplicidades se agregan en un extremo de la relación puesto que definen en si misma un rol, a pesar que este no tenga nombre.

En la siguiente figura se pueden apreciar varias asociaciones que definen distinta multiplicidades



De esta manera, el diagrama especifica que existe una asociación llamada “tiene” entre Auto y Motor del tipo “Auto-tiene-Motor” donde por cada instancia de Auto existe una instancia de Motor.

En cada rol se pueden definir distintas multiplicidades y sus significados son como los que muestra la siguiente figura:



Ejemplo

Para el siguiente ejemplo se presentará un análisis básico del enunciado de un problema y el posterior análisis de objetos candidatos para realizar un diagrama de objetos. El fin de este ejemplo es exponer parte del diseño de solución pero no así todo el proceso involucrado en un análisis y diseño orientado a objetos

Enunciado del problema:

Se debe manejar las asignaciones de cursos en un instituto de enseñanza informando a instructores y alumnos las asignaciones de curso que tienen designadas. Se cuenta, para esto, con las direcciones de ambos con los datos completos. Cada alumno puede ser asignado como mínimo a un curso y los cursos deben tener un mínimo de tres alumnos para que se coloquen en el calendario y comiencen

Por otra parte, los cursos son asignados según un calendario en el cual pueden tener más de un horario (el mismo curso en diferentes horarios). Cada curso es asignado a un aula las cuales tiene diferentes cantidades de escritorios, sillas, pizarrones y computadoras.

Determinación del contexto del problema

El problema a resolver sólo tiene en cuenta el manejo para las asignaciones de cursos, aulas, alumnos e instructores. Todo manejo más allá de esto se analiza en la resolución de otros problemas enunciados.

Lista de objetos candidatos:

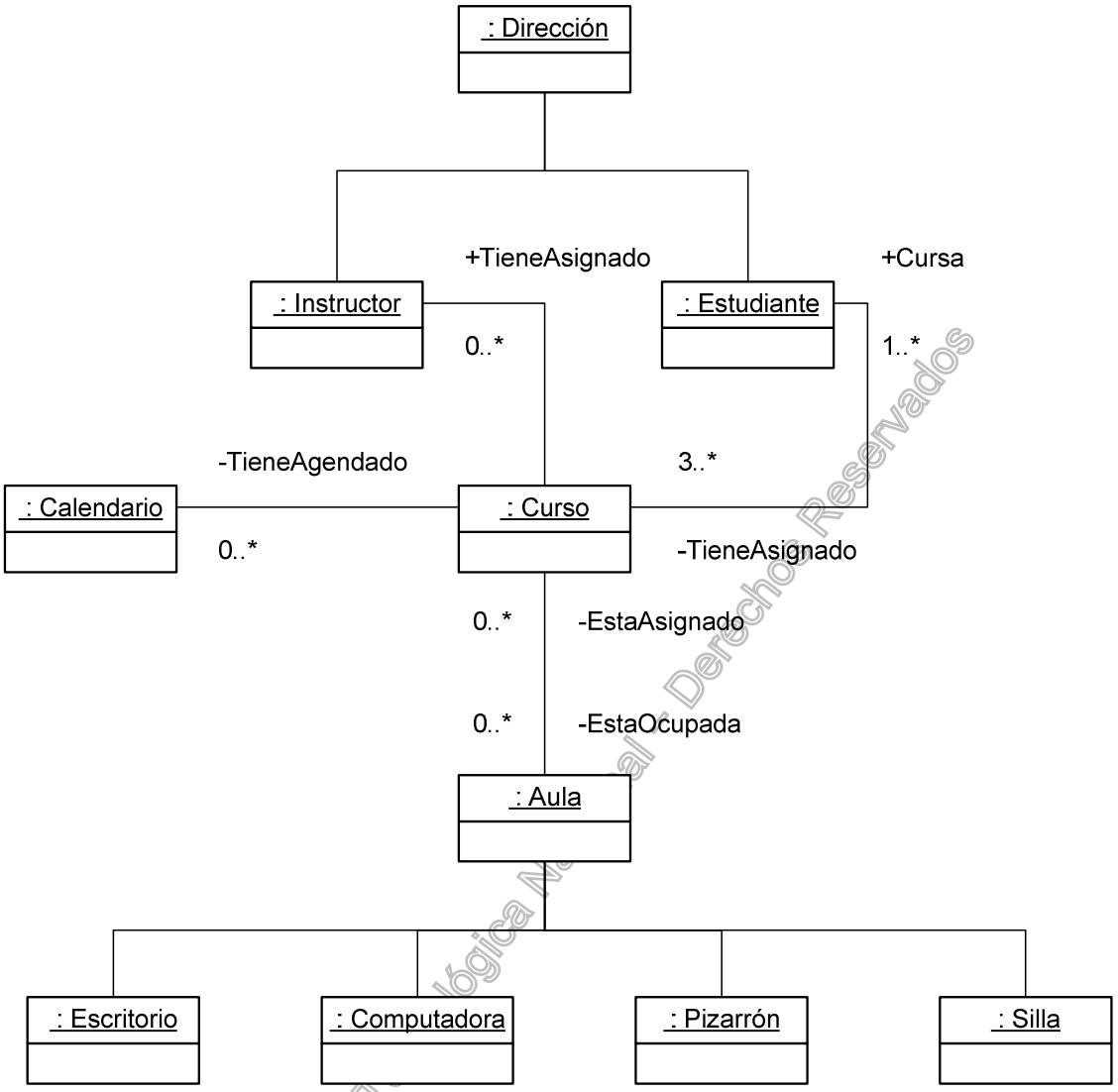
- Instituto de enseñanza
- Instructor
- Alumno

- Curso
- Calendario
- Horario
- Aula
- Escritorio
- Silla
- Pizarrón
- Computadora

Análisis de los objetos encontrados

Objetos Candidatos	Descripción	Estado	Motivo
Instituto de enseñanza	Es la entidad en donde se dictan los cursos a asignar	Descartado	Esta fuera del contexto las particularidades del Instituto
Instructor	Es quien dicta el curso	Activo	Es parte del contexto
Alumno	Es quien asiste al curso	Activo	Es parte del contexto
Dirección	Lugar de residencia	Descartado	Es un atributo
Curso	Es la materia a dictar	Activo	Es parte del contexto
Calendario	Es la organización de horarios para el dictado de los cursos	Activo	Es parte del contexto
Horario	Es el tiempo en el cual se pueden asignar un o más cursos	Descartado	Es el atributo del calendario que define el rango de validez del mismo
Aula	Es el lugar físico donde se dictan los cursos	Activo	Es parte del contexto
Escritorio	Es donde trabajan los alumnos y el instructor	Activo	Es parte del contexto
Silla	Es donde se sientan los alumnos y el instructor	Activo	Es parte del contexto
Pizarrón	Es la herramienta sobre la que se escribe para la clase cuando es necesario	Activo	Es parte del contexto
Computadora	Es la herramienta utilizada para las prácticas y o seguimientos de las clases por los alumnos e instructores	Activo	Es parte del contexto

Diagrama de colaboración entre objetos



Ejercicio 1

Los temas de los que se tratan en este ejercicio son los siguientes:

- Asociaciones
- Multiplicidad

Dado un enunciado de problema, se debe realizar un análisis y diseño (preliminar) en base a este que refleje asociaciones y multiplicidad

Asociaciones complejas

Existen situaciones en las cuales las clases se relacionan con otras por medio de multiplicidades en ambos roles de la asociación que no se pueden determinar en el momento del diseño. Una causa puede ser que en ambos lados de la asociación pueden existir múltiples instancias de objetos del tipo de la clase que interviene en la relación.

En lo diagramas UML, se identifican cuando los demarcadores de la multiplicidad no definida (el "*") aparece en ambos lados de la relación (en términos de bases de datos, son relaciones de muchos a muchos).

Este tipo de situaciones requieren una forma de manejar la asociación de manera que pueda ser fácilmente identificada cada relación. Para ello, se descompone una relación de muchos a muchos en dos relaciones, una de muchos a uno y otra de uno a muchos.

Sin embargo, en el medio de esta descomposición debe existir un mecanismo que las gestione. Este último presenta un nuevo problema que se puede resolver de dos maneras diferentes. Una de ellas es utilizando una clase de asociación, la cual gestionará ambos extremos de cada relación uno a muchos. La otra, como se verá posteriormente, se llama asociación calificada.

Cuando existe una relación del tipo:

> *Una persona puede reservar muchos libros en una biblioteca. A la vez, un mismo libro puede ser reservado por muchas personas.*

Se identifica claramente una asociación de "muchos a muchos", la cual se puede diagramar como muestra la siguiente figura:

Se debe notar que en cada rol de la asociación se identifica la multiplicidad con un "*".

Clase de asociación

Como se mencionó anteriormente, cuando aparecen asociaciones complejas, se necesita un mecanismo que las maneje. Una técnica para manejar asociaciones complejas es intercalar en la relación, una clase dedicada a manejarla. De esta manera, la clase se utiliza para resolver las relaciones de muchos a muchos.

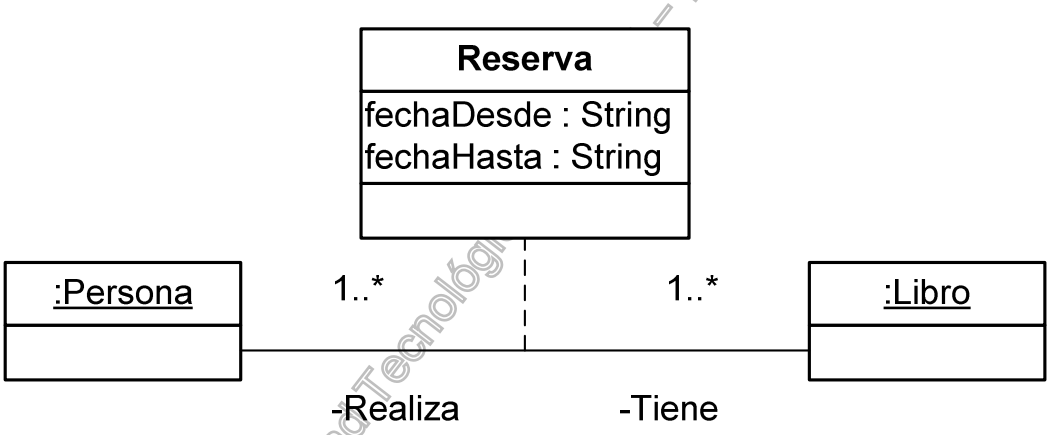
Sin embargo, la existencia de una clase de este tipo es dedicada a resolver la relación, por lo tanto sus métodos y atributos están dedicados a resolver el conflicto y es una buena práctica de diseño que este sea su único fin.

Volviendo al enunciado previo del problema de diseño a resolver:

> *Una persona puede reservar muchos libros en una biblioteca. A la vez, un mismo libro puede ser reservado por muchas personas.*

La primera solución dada por el diagrama

Se puede reformular como:



Lo cual permite el manejo de la relación.

Asociación calificada

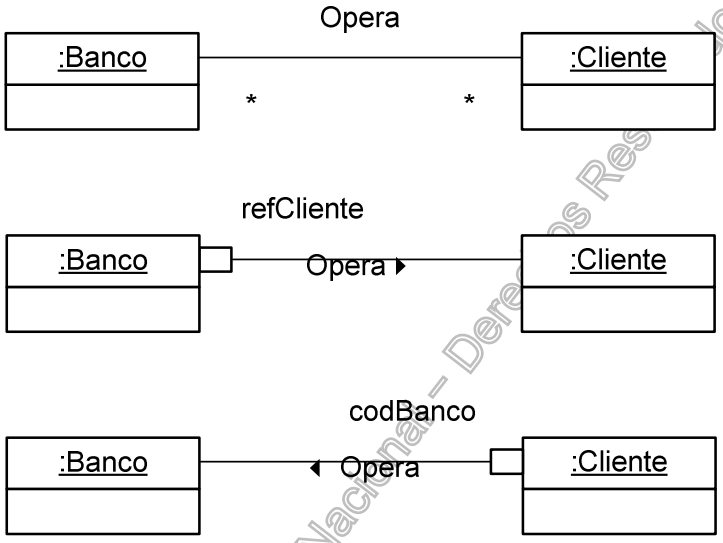
Las clases de asociación no son la única técnica que existe para resolver este tipo de asociaciones. Por ejemplo, se puede tener un elemento que pertenezca a una clase que realice la misma función que la clase de asociación un extremo de la relación para cada clase. La salvedad, es que el rol en cada extremo es más complejo que un rol común y se define y diagrama diferente.

Un ejemplo en la codificación, es si la clase tiene un vector o una colección de objetos entre sus atributos para manejar la relación.

El atributo deberá almacenar o tener la capacidad de referenciar los elementos individuales de la relación compleja.

Suponiendo el siguiente ejemplo:

> *Un banco opera con muchos clientes, pero estos, a su vez, pueden operar con muchos bancos*



Una solución posible es la que se muestra en el diagrama

Los atributos definidos, refCliente y codBanco, manejan múltiples instancias de la clase con la que se relaciona. Estos atributos almacenan la referencia pero la clase que los posee debe proveer un mecanismo para acceder a dichos elementos.

Ejercicio 2

El temas del que trata este ejercicio es el siguiente:

- Asociaciones Complejas

En base al diagrama realizado en el ejercicio anterior, analizar las relaciones complejas y proponer un diseño adecuado

Agregaciones

Es una relación más fuerte entre clases. Por lo general, cuando se enuncia el problema de diseño se puede reconocer fácilmente una agregación porque se caracteriza por las palabras "**tiene un**" o "**tiene una**". A esta frase se la llama "frase de validación" y es la que la determina.

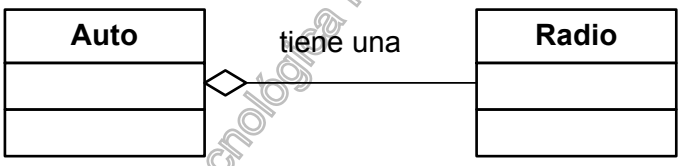
Otra forma de reconocer las asociaciones es la relación "totalidad – parte de". Por ejemplo, se puede definir como problema de diseño que:

Un auto tiene una radio

El auto es la totalidad y la radio es una parte que se reconoce como un objeto independiente. Sin embargo, es claro que aunque la relación puede interpretarse como "totalidad – parte de", esto no implica que ambos objetos mantengan su relación durante todo el tiempo que cada uno este activo (por ejemplo, un auto puede cambiar el algún momento la radio que tiene), aunque también, la relación puede durar la vida del objeto (por ejemplo, un auto nunca cambia la radio que tiene aunque pueda).

Los diagramas que representan las agregaciones se dibujan con un rombo en blanco del lado de la clase que representa la "totalidad" y como es un tipo de relación, todo lo expuesto para las relaciones anteriormente (nombres, enlaces, roles, multiplicidad) es válido también para ellas.

Un diagrama que representa la agregación del problema de diseño antes enunciado, es el siguiente:



Composiciones

Es la relación más fuerte entre clases, ya que cuando una clase compone un objeto del tipo de otra, el problema de diseño indica que para resolverlo, una clase no tiene sentido sin la instancia del objeto de la clase que compone.

La frase de validación se caracteriza por las palabras "**siempre tiene**". Por ejemplo, un problema de diseño donde se puede apreciar lo afirmado es el siguiente:

Un auto siempre tiene un motor

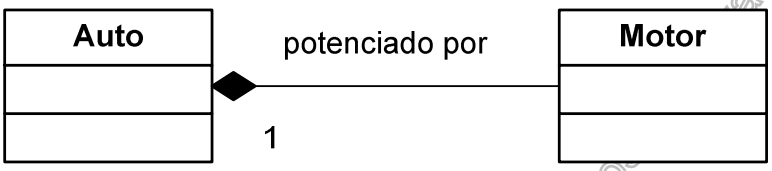
Una factura de siempre tiene detalles de facturación

Las composiciones tiene como característica que duran el mismo tiempo que la vida del objeto siempre, ya que si faltase el objeto compuesto, el que lo compone no tiene sentido. Por ejemplo,

no tiene sentido tratar de diseñar un auto sin su motor o una factura sin detalles de facturación (en este caso la composición también incluye una asociación compleja).

Los diagramas que representan las composiciones se dibujan con un rombo en negro del lado de la clase que representa la "totalidad" y como es un tipo de relación, todo lo expuesto para las relaciones anteriormente (nombres, enlaces, roles, multiplicidad) es válido también para ellas.

Un diagrama que representa la composición del problema de diseño antes enunciado, es el siguiente:



Ejercicio 3

Los temas de los que se tratan en este ejercicio son los siguientes:

- Agregación
- Composición

En base al diagrama realizado en el ejercicio anterior, analizar las relaciones las agregaciones y composiciones para proponer un diseño adecuado.