**Modelado con UML**

**Ejemplo como abordar la generación de un diagrama de clases**

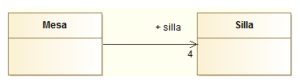
https://joanpaon.wordpress.com/2013/07/01/uml-diagrama-de-clases-ejercicio-1/

# **Diagramas de Clases – Relaciones**

Las **relaciones** son el **tercer pilar fundamental** en el que se basan los Diagramas de Clases, después de las **clases** mismas y los **interfaces**. Las relaciones **se aplican exclusivamente entre clases** y pueden ser **binarias o de orden superior**. Decir que dos clases están relacionadas entre si viene a significar que esas clases tienen algo que ver entre sí. De cómo sea la **naturaleza de la relación** definirá un tipo u otro de vinculación. De lo que se trata aquí es de identificar, caracterizar y ejemplarizar cada una de ellas.

## Asociación

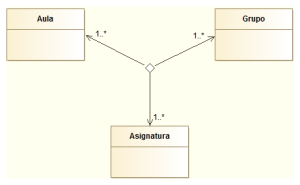
La forma más sencilla de relación es aquella denominada **asociación**. La asociación se utiliza para expresar simplemente que dos clases están vinculadas entre sí. En ella se expresa la **navegabilidad** entre la **clase origen** y la **clase destino**, y la **cardinalidad** de la **clase destino** en la asociación.

[](https://joanpaon.files.wordpress.com/2013/06/5.png)

El Diagrama de Clases del ejemplo anterior permite representar, en el contexto de UML,  el hecho de que una mesa tiene 4 sillas a juego.  Obsérvese que en el diagrama no se expresa más vinculación que la navegabilidad que expresan que las sillas van con la mesa, ni más restricción que la cardinalidad que expresa que con la mesa van cuatro sillas.

En la figura se observa como aparece el rol **silla** para vincular la clase **Silla** a la clase**Mesa**. Esta vinculación se sustanciará convirtiendo el **rol de la relación** en **un atributo de la clase origen que referencia la clase destino**.

Estrictamente hablando una asociación no tiene que ser únicamente de navegabilidad en un solo sentido, Puede ser en ambos con los que ambas clases son origen y destino a la vez. Un tipo especial de esta situación acontece cuando la asociación involucra más de dos clases. En ese caso todas las clases asociadas son origen y destino a la vez.

[](https://joanpaon.files.wordpress.com/2013/06/6.png)

El ejemplo anterior se modeliza la siguiente situación.

*Cada aula alberga uno o más grupos a los que se imparten una o más asignaturas, a su vez cada grupo tiene asignada una o más aulas en donde recibe docencia de una o más asignaturas, y además cada asignatura se imparte en una o más aulas a uno o más grupos.*

En las asociaciones, el peso de la definición de la relación recae enteramente sobre la parte [TODO]:

* La vinculación de define en la parte [TODO] que incluye la referencia a la parte [PARTE].
* La multiplicidad de  la parte [PARTE] debe expresarse en la parte [TODO] a través de algún tipo de colección, a la cual debe de acompañar, en la mayoría de los casos, al menos de sendos métodos para incorporar/desvincular elementos.

El concepto de asociación entre clases permite representar la semántica de muchas situaciones. Sin embargo la realidad ofrece situaciones mucho más complejas cuyo modelizado exige evolucionar este concepto de asociación introduciendo el **concepto de pertenencia** y el **concepto de autonomía**.

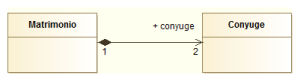
## Pertenencia

Para explicar la semántica de pertenencia de una relación ayuda el plantearla desde un punto de vista de binomio **[PARTE] – [TODO]**. Desde esta perspectiva una relación, básicamente binaria, está constituida por un componente **[PARTE]** y un componente**[TODO]**.

El componente **[PARTE]** se caracteriza porque es una pieza, en el sentido constructivo, del componente **[TODO]**. El componente **[TODO]** tiene la capacidad de albergar al componente **[PARTE]** integrándolo dentro de sí mismo.

Antes de seguir un buen ejemplo ayudaría a fijar los conceptos de **[PARTE]** y **[TODO]** en una relación entre clases.

Bien, considérese el ejemplo de la relación de un **matrimonio** respecto de sus **cónyuges**. En esa relación el **matrimonio** seria la parte **[TODO]**, mientras que los **cónyuges** serian la parte **[PARTE]** de la relación. Trasladandando el ejemplo al contexto UML, si se considera la clase **Matrimonio** y la clase **Conyuge**, y dejando para más adelante la explicación de los detalles involucrados en la relación, el **Diagrama de Clases** que representaría esta relación podría ser el siguiente:

[](https://joanpaon.files.wordpress.com/2013/06/9.png)

Otro ejemplo. Considérese la relación que existe entre los **operarios de una fábrica** y las **secciones de trabajo** de la misma. En este caso cada operario trabaja en un momento dado en una sección de la fábrica, aunque después puede trabajar en otra. Así pues, cada sección de la fabrica alberga un número determinado de operarios.

[](https://joanpaon.files.wordpress.com/2013/06/7.png)

## Autonomía

De lo que se trata de dilucidar en este apartado es, qué **ciclo de vida** tienen los objetos de la **clase [TODO]** y qué **ciclo de vida** tienen los objetos de la **clase [PARTE].**Dependiendo de cómo se concreta esta cuestión la semántica de la situación define un tipo de asociación u otro entre las respectivas clases.

En concreto, la **regla para determina el tipo de asociación** es fijarse en el **ciclo de vida** de los objetos de la **clase [TODO]**, en concreto en el **momento en que se destruye**. La pregunta que hay que hacerse es **¿Qué ocurre con los objetos de la clase [PARTE]?**. La respuesta a esta pregunta determina **dos tipos de asociaciones**:

* **Agregación**. Cuando el objeto [TODO] se destruye, los objetos [PARTE] pueden seguir existiendo autónomamente.
* **Composición**. Cuando el  objeto [TODO] se destruye también desaparecen los objetos [PARTE], cuya existencia ya no tiene sentido.

## Agregación

Es un tipo de asociación en donde el ciclo de vida de la parte [TODO] está desvinculado del ciclo de vida de la parte [PARTE], de tal manera que cuando desaparece la parte [TODO] la parte [PARTE] puede seguir existiendo. A este tipo de vinculación se la denomina también **asociación débil** o **asociación funcional**.

Para ejemplarizar este tipo de relación considérese el caso expuesto anteriormente respecto de los operarios y las secciones de una fábrica.

[](https://joanpaon.files.wordpress.com/2013/06/7.png)

En el ejemplo, la clase **Seccion** referencia las instancias de la clase **Operario**, que se corresponden con los operarios que están trabajando en ella.

Tomando como referencia el ejemplo anterior, se inferirán las correspondientes reglas respecto a las agregaciones en los diagramas de clases:

* La **clase [TODO]** se identifica con un **rombo en blanco**.
* La **clase [PARTE]** se identifica con una **flecha de navegación**.
* La **relación** se identifica por su **rol situado en la clase [PARTE]**.
* La **clase [TODO] no tiene un atributo para expresar el rol**.
* La <strong”>multiplicidad de la clase [TODO] es diferente de la unidad.
* El **constructor** de la clase [TODO] **no instancia** la clase [PARTE].
* El **destructor** de la clase [TODO], cuando existe, **no altera** la clase [PARTE].

La codificación del Diagrama de Clases del ejemplo anterior en Java podría corresponderse con el siguiente código:

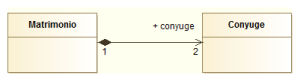
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | class Operario {     ...  }  class Seccion {     ...     ArrayList<Operario> listaOperarios = new ArrayList<Operario>();     ...  } |

Como se puede observar en el código anterior la multiplicidad de la clase [TODO] no se expresa porque depende de las restricciones funcionales de la clase que instancia la parte [TODO].

## Composición

Es un tipo de asociación en donde el ciclo de vida de la parte [PARTE] está vinculado del ciclo de vida de la parte [TODO], de tal manera que cuando desaparece la parte [TODO] la parte [PARTE] también desaparece. A este tipo de vinculación se la denomina también **asociación fuerte**o **asociación existencial**.

Para ejemplarizar este tipo de relación considérese el caso expuesto anteriormente respecto de los cónyuges y el matrimonio.

[](https://joanpaon.files.wordpress.com/2013/06/9.png)

En el ejemplo, la clase **Matrimonio** referencia cada una de las dos instancias de la clase **Conyuge**, generalmente a través de algún tipo de documento en el registro civil.

Tomando como referencia el ejemplo anterior, se inferirán las correspondientes reglas respecto a las agregaciones en los diagramas de clases:

* La **clase [TODO]** se identifica con un **rombo en negro**.
* La **clase [PARTE]** se identifica con una **flecha de navegación**.
* La **relación** se identifica por su **rol situado en la clase [PARTE]**.
* La **clase [TODO] no tiene un atributo para expresar el rol**.
* La **multiplicidad de la clase [TODO] es siempre la unidad**.
* El **constructor** de la clase [TODO] **suele instanciar** la clase [PARTE].
* El **destructor** de la clase [TODO], cuando existe, **destruye también** la clase [PARTE].

La codificación del Diagrama de Clases del ejemplo anterior en Java podría corresponderse con el siguiente código:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | class Conyuge {     ...  }  class Matrimonio {     ...     Conyuge[] conyuge = new Conyuge[2];     ...     Matrimonio() {        conyuge[0] = new Conyuge();        conyuge[1] = new Conyuge();        ...     }  } |

Al igual que en la agregación la multiplicidad de la clase [TODO] no se expresa porque depende de la restricciones funcionales de la clase que instancia la parte [TODO].

**Ejercicios partiendo del enunciado se debe hacer:**

1. Hacer el Diagrama de casos de uso del siguiente enunciado:
2. Hacer un diagrama de secuencia identificado del enunciado anterior
3. Hacer un diagrama de estados
4. Hacer un diagrama de componentes
5. Hacer el modelo entidad relación

Una cadena de agencias de viajes desea disponer de una Base de Datos que contemple información relativa al hospedaje y vuelos de los turistas que la contratan.

Los datos a tener en cuenta son:

La cadena de agencias está compuesta por un conjunto de sucursales. Cada sucursal viene definida por el código de sucursal, dirección y teléfono.

La cadena tiene contratados una serie de hoteles de forma exclusiva. Cada hotel estará definido por el código de hotel, nombre, dirección, ciudad, teléfono y número de plazas disponibles.

De igual forma, la cadena tiene contratados una serie de vuelos regulares de forma exclusiva. Cada vuelo viene definido por el número de vuelo, fecha y hora, origen y destino, plazas totales y plazas de clase turista de las que dispone.

La información que se desea almacenar por cada turista es el código de turista, nombre y apellidos, dirección y teléfono.

Por otra parte, hay que tener en cuenta la siguiente información:

A la cadena de agencias le interesa conocer que sucursal ha contratado el turista.

A la hora de viajar el turista puede elegir cualquiera de los vuelos que ofrece la cadena, y en que clase (turista o primera) desea viajar.

De igual manera, el turista se puede hospedar en cualquiera de los hoteles que ofrece la cadena, y elegir el régimen de hospedaje (media pensión o pensión completa). Siendo significativa la fecha de llegada y de partida.