

MP_0487. Entornos de desarrollo

UF5. Lenguaje unificado de modelado UML

5.2. Los principales tipos de diagramas

Índice

☰	Objetivos	3
☰	Casos de uso	4
☰	Actividad casos de uso I	8
☰	Actividad casos de uso II	10
☰	Las clases	11
☰	Visibilidad de los atributos	14
☰	Las relaciones entre clases	16
☰	Relación de asociación	18
☰	Multiplicidad de las asociaciones	20
☰	Clases enumeradas	21
☰	Actividad diagramas de clases I	22
☰	Actividad diagramas de clases II	24
☰	Diagrama de estados	26
☰	Diagrama de actividad	28
☰	Diagramas de secuencia	29
☰	Diagramas de colaboración	30
☰	Diagramas de objetos	32
☰	Diagrama de paquetes	34
☰	Diagramas de componentes	35
☰	Diagramas de despliegue	36
☰	Resumen	37

Objetivos

Con esta lección perseguimos los siguientes objetivos:

1

Comprender la utilidad de los diagramas de diseño.

2

Definir qué es el lenguaje de modelado UML.

3

Estudiar los principales diagramas de UML para el diseño de software.

¡Ánimo y adelante!

Casos de uso

Los diagramas de casos de uso son importantes para modelar el comportamiento de un sistema, un subsistema o una clase.

Están formados por tres elementos: **actores** (quién interacciona con el sistema), casos de uso (lo que hace el sistema) y sus **relaciones** (interacciones entre actores y sistema).



Ejemplos casos de uso.

Las relaciones entre casos de uso y/o actores son las siguientes:

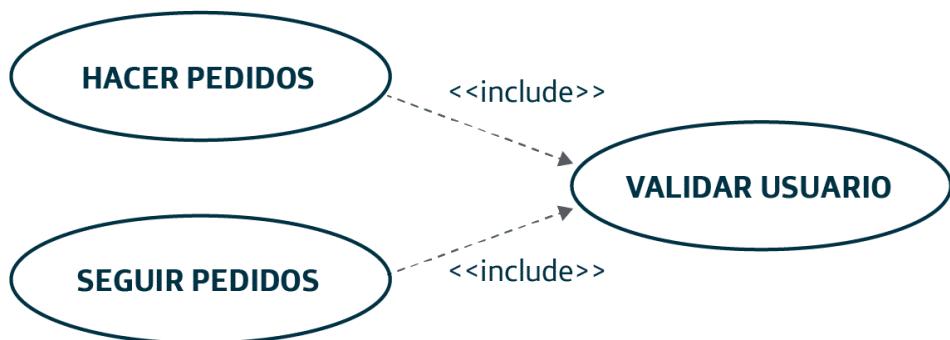
Comunicación

Relación entre un actor y un caso de uso con el que interactúa. Se representa con una línea. Uso (*include, use*):



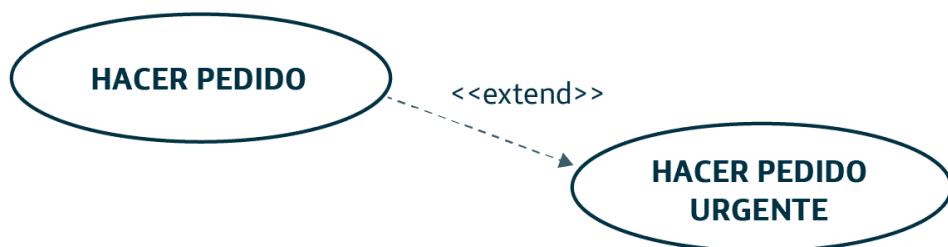
Relación entre casos de uso

Se representa por una línea discontinua con una flecha en el sentido de la relación.



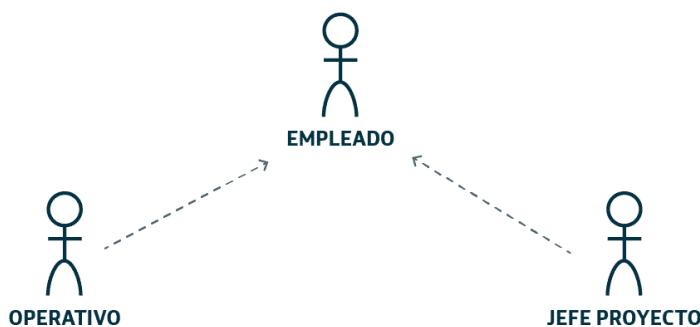
Extensión (extend)

Relación entre casos de uso. Se representa por una línea discontinua con una flecha en el sentido de la relación.



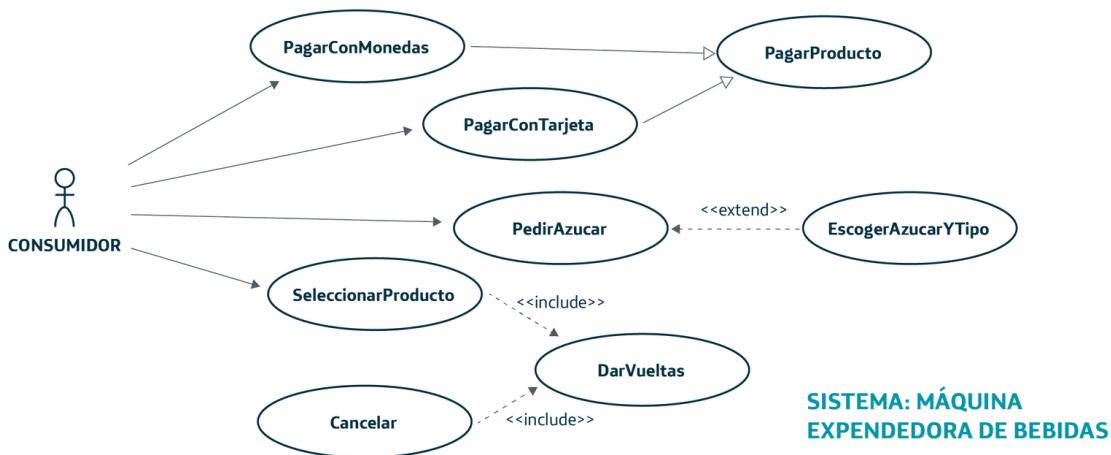
Generalización

Relación entre casos de uso y entre actores. Expresa el mismo concepto de herencia que existe entre clases. Se representa por una línea continua con un triángulo vacío señalando el sentido de la relación.



Ejemplo

Veamos un ejemplo de un **diagrama de casos de uso** para un sistema que representa una máquina expendedora de bebidas (café, té, etc.), con la que se puede pagar en efectivo o con tarjeta monedero.



Ejemplo de caso de uso de una máquina expendedora.

Actividad casos de uso I



Vamos a plantear una actividad en la que tenemos que crear un diagrama de casos de uso.

Así que coge bolígrafo, papel y ¡adelante!

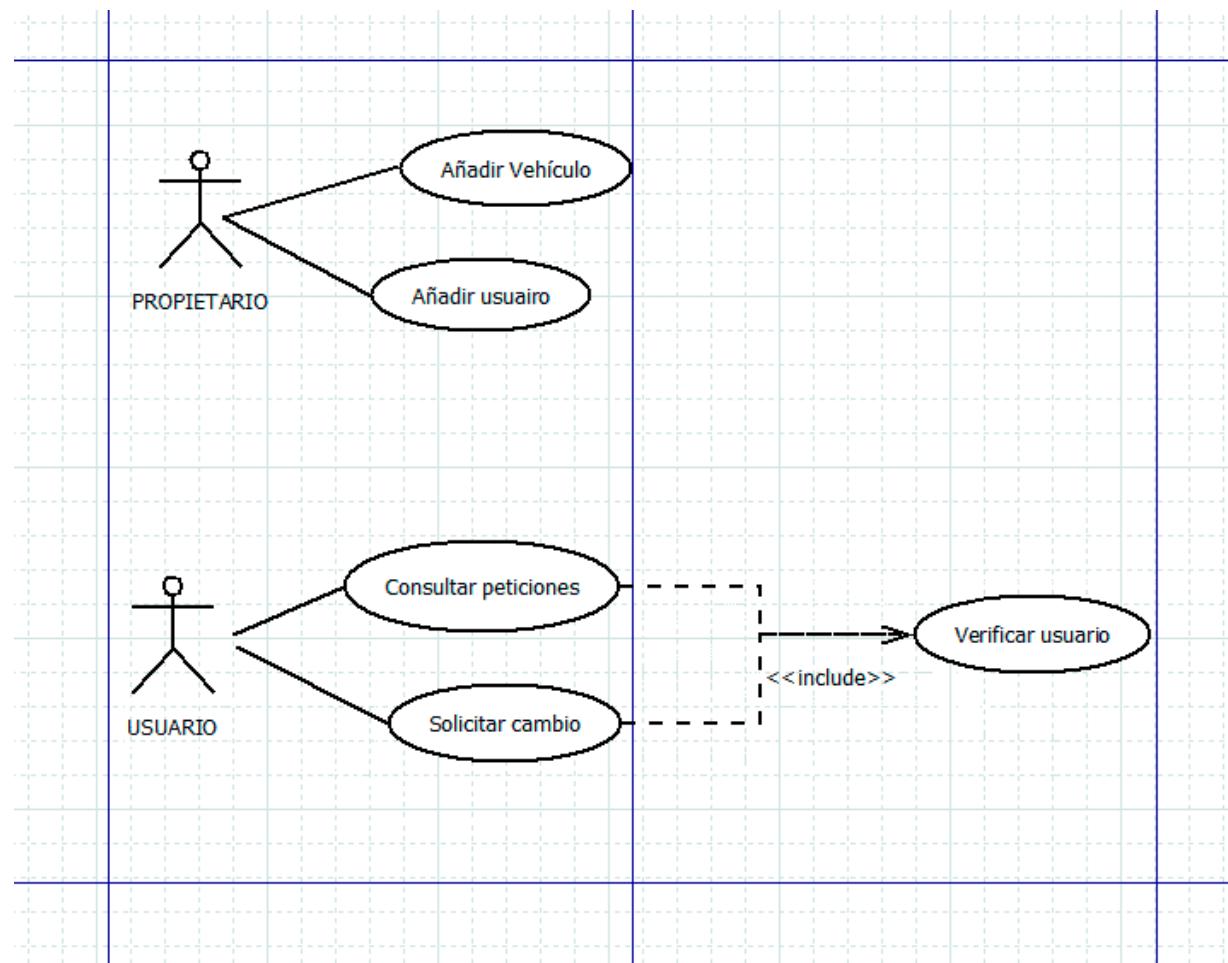
Imagina que queremos crear un sistema para gestionar el uso de coches compartidos entre usuarios. Tenemos a los propietarios de los vehículos, que dan de alta un vehículo y seleccionan los usuarios que podrán hacer uso del mismo.

Los usuarios pueden solicitar cambiar de vehículo cuando quieran, pudiendo consultar las peticiones de cambios de otros usuarios.

Los usuarios solo podrán consultar y pedir cambios de los vehículos a los que el propietario le diese permiso.

¿Cómo sería el diagrama de casos de uso?

Solución



Actividad casos de uso II

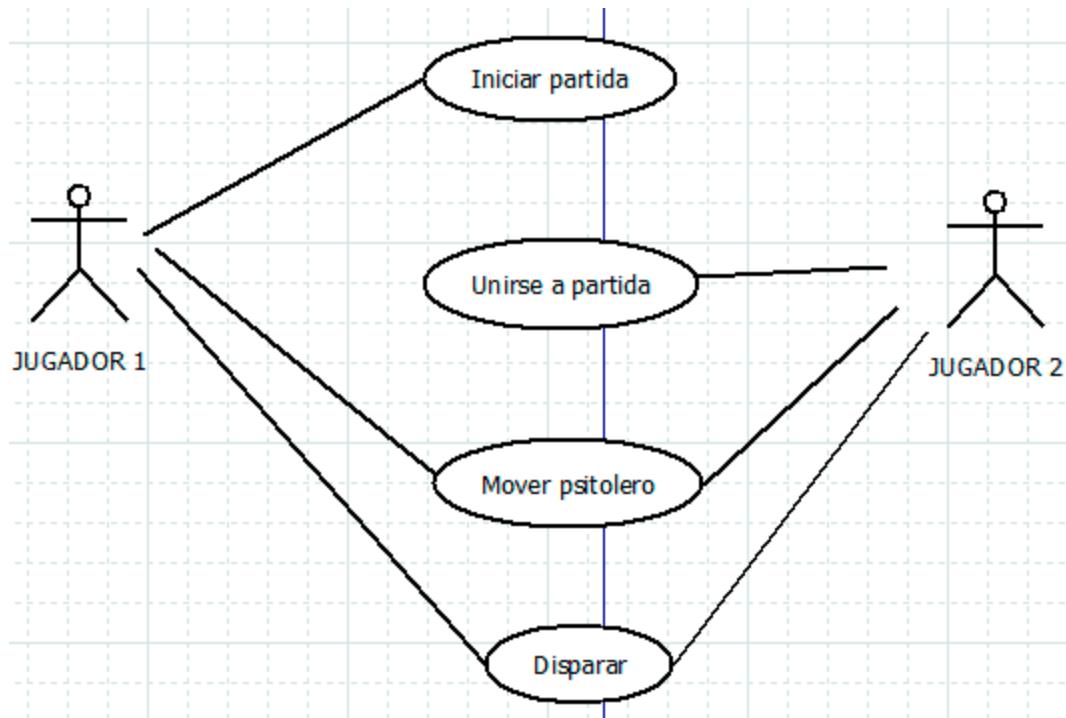
Continuemos con otro ejercicio.

¡No sueltes el bolígrafo!

Ahora queremos crear un juego donde participan dos jugadores *on-line*. Cuando desean jugar uno de ellos tiene que crear la partida, mientras que el otro se une a ella. El objetivo del juego es disparar al contrario. Si uno de los dos jugadores acierta, la partida termina.

¿Cómo sería el diagrama de casos de uso?

Solución



Las clases

Una **clase** es un descriptor de objetos que comparten los mismos **atributos**, operaciones, **métodos**, relaciones y comportamientos.

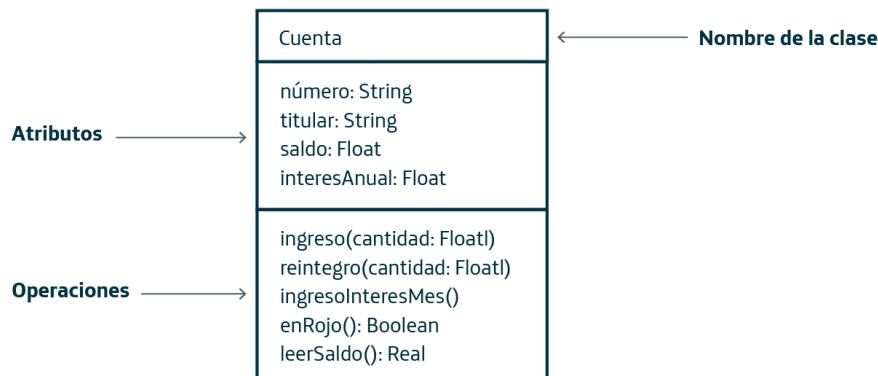
Las clases representan las entidades de un programa, una factura, un producto, una persona...

Como toda **entidad**, tiene un nombre que define su tipo, unos atributos que determinan su estado interno dentro de la clase, y unos métodos que son las acciones que pueden realizar.

Una clase se representa con un rectángulo con tres compartimentos para:

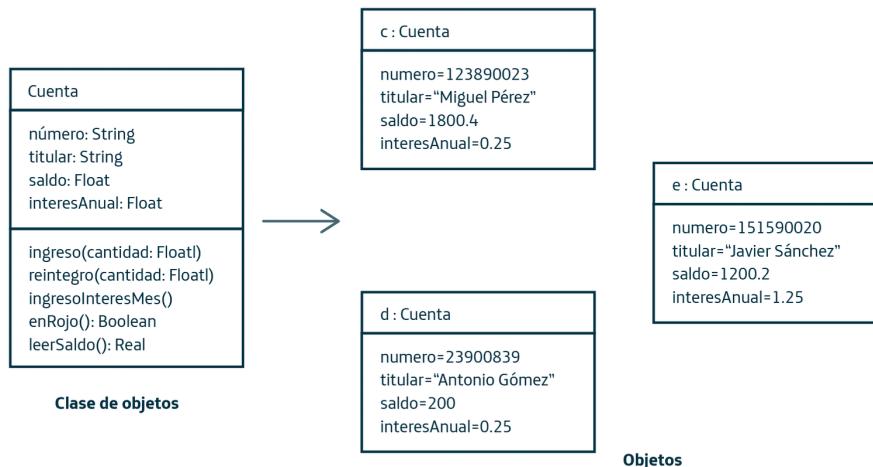
- Nombre.
- Atributos.
- Operaciones.

Solo es obligatorio el compartimento para el nombre.



Partes de una clase.

Cuando un programa se ejecuta se produce la instanciación de los objetos. Las clases generan los objetos representando cada uno con sus características propias (los atributos).



Representación de una clase instanciando objetos.

Esta representación describe la información necesaria para la construcción del código. Por ejemplo, si quisieramos codificar la clase representada sería:

```
class Cuenta {  
  
    long numero;  
    String titular;  
    float saldo;  
    float interesAnual;  
  
    void ingreso(float cantidad) {}  
    void reintegro(float cantidad) {}  
    void ingresoInteresMes() {}  
    boolean enRojos() {}  
    float leerSaldo() {}  
}
```

Codificación en Java de la representación del objeto.

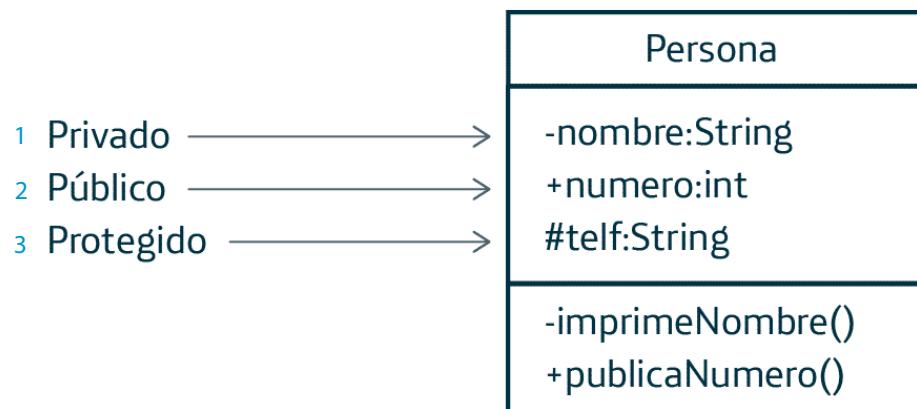
Algunas consideraciones a tener en cuenta:

- 1 Cuando se dibuja una clase **no es obligatorio mostrar ni todos los compartimentos, ni todos los atributos, ni todas las operaciones.**
- 2 El contexto donde aparece la clase **debe ser significativo y contribuir a que el dibujo sea claro y ayude a comprender su utilidad.**
- 3 Un compartimento vacío no significa que la clase no tenga atributos u operaciones, simplemente se ha decidido no mostrarlos.
- 4 Para indicar que hay más atributos u operaciones en un compartimento se termina la lista con puntos suspensivos.
- 5 Para organizar las listas largas de atributos u operaciones se pueden utilizar estereotipos con un nombre que clarifique la agrupación resultante.

Un **diagrama de clases** es una representación gráfica de la vista estática del conjunto de clases que forman parte de un sistema, **junto con las relaciones existentes** entre ellas.

Visibilidad de los atributos

Existen tres niveles de visibilidad para los atributos y las operaciones:



1 -privado

Visible solo para la clase.

2 + público

Visible a todos los clientes de la clase y a los objetos instanciados.

3 # protegido

Visible a las clases que hereden de ella.

```
class Persona {  
  
    private String nombre;  
    public int numero;  
    protect String telf;  
  
    private void imprimeNombre() {  
        ....  
    }  
    public void publicaNombre() {  
        ....  
    }  
}
```

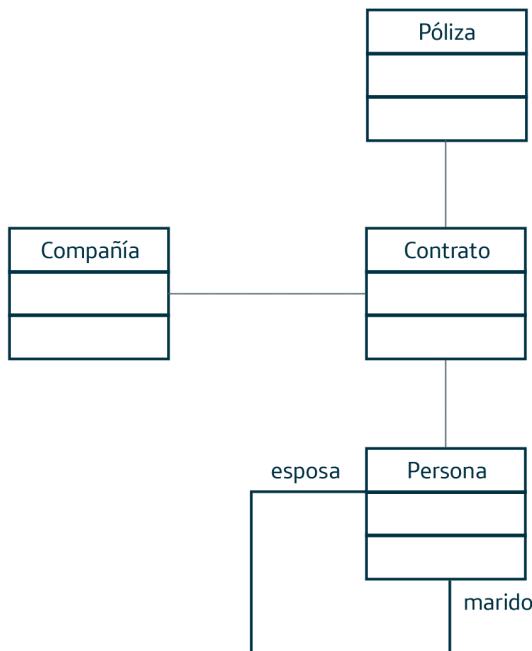
Código en Java de la clase.

Las relaciones entre clases

En un diagrama de clases básicamente se pueden dar tres tipos de relaciones:

Relación de asociación

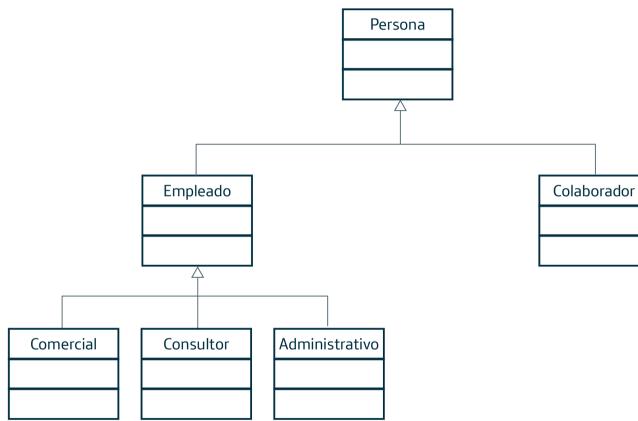
Es una **relación entre varias clases**, incluso puede estar relacionada consigo misma. Los objetos en muchas ocasiones están relacionados por algunos atributos, como puede ser un cliente a un pedido y/o a una factura, etc.



Relación de asociación.

Relación de generalización (herencia)

Es la relación de clases donde **una entidad más general se divide en otras más específicas**. Son las relaciones de herencia, donde el elemento padre posee los atributos y métodos que heredan todos sus descendientes, mientras que los hijos poseen atributos únicos respecto al padre.



Ejemplo de clases con herencia.

Relación de dependencia

Relación entre dos clases en las que un elemento puede afectar el estado de otro. Esta es la relación menos importante de las comentadas. Simplemente refleja que la implementación de una clase depende del estado de la otra.



Ejemplo relación de dependencia.

Relación de asociación

Como vimos en el punto anterior, la relación de asociación indica que dos clases se relacionan entre sí.

Si la relación es normal (los objetos no dependen de la existencia del otro) puede representarse por una línea simple.



Asociación normal.

Pero no todas las asociaciones son iguales, en algunos casos **la existencia de un objeto depende de otro**. Es decir, una ventana tiene un marco, bisagras... Estos objetos componen el objeto principal y, a diferencia de las herencias, son objetos diferentes.

Para diferenciar este tipo de asociaciones existe la agregación o la composición.

Agregación



Implica que una de las clases es el "todo" y la otra "es parte". Este tipo de asociación no obliga a que objetos de las dos clases tengan existencia dependiente una de la otra.



Composición

A diferencia de la agregación, esta relación sí obliga a la existencia de la otra clase. No puede existir de forma independiente.

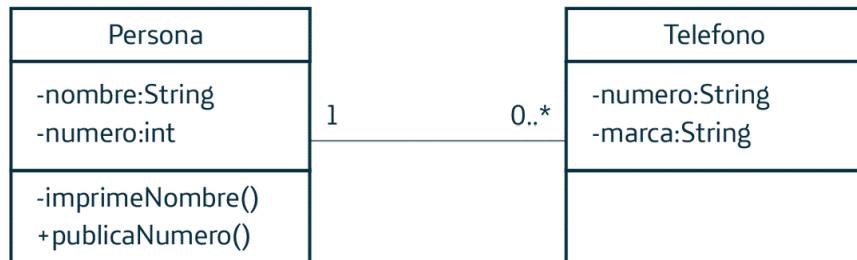
Multiplicidad de las asociaciones

El **número de instanciaciones de una clase** se **relacionan con una instancia** de otra clase.

La multiplicidad de una asociación determina el número de objetos que, instanciados desde una clase, se relacionan con otra. Toda asociación tiene dos multiplicidades, una en cada extremo de la misma.

Por ejemplo, un cliente compra varios productos, pero un producto no puede ser comprado por más de un cliente.

Para indicar la multiplicidad de una relación tenemos que indicar un mínimo y un máximo.



Una persona puede tener ninguno o indefinidos teléfonos.

Las multiplicidades que podemos representar son:

1 Uno y solo uno.

* Cero o varios.

0..1 Cero o uno.

1..* Al menos uno.

Clases enumeradas

Un enumerado (o *Enum*) es una clase en la que solo podemos crear objetos con una especificación determinada. Los enumerados, respecto al resto de clases, sí tienen constructor, debe ser privado y no permitirá la creación de objetos no especificados en la misma.

En UML estas clases no suelen estar relacionadas directamente, pero se representan en el diagrama si se considera necesario.



Representación clase enumerada.

```
public enum Tipo {
    SOBREMESA, PORTATIL, TODOENUNO, MINIPC
}
```

En la mayoría de las ocasiones este tipo de propiedades se puede representar también con una relación de dependencia.

Actividad diagramas de clases I

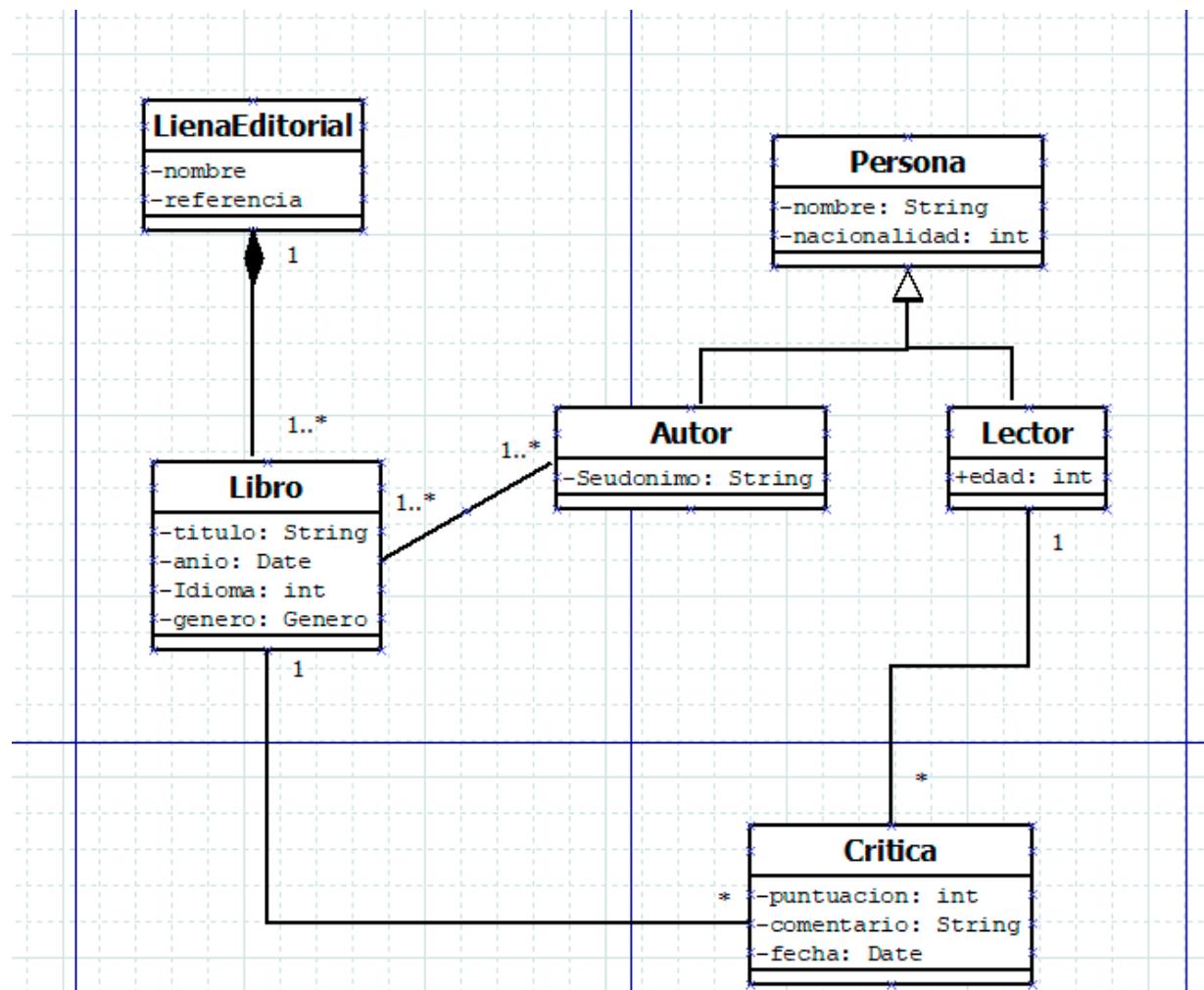
Vamos a hacer otro pequeño ejercicio, en este caso vamos a plantear un diagrama de clases.

Ya sabes, necesitas boli, papel y ¡a pintar!

Representa mediante un diagrama de clases el siguiente sistema para gestionar libros:

- Los libros se caracterizan por su título, año de publicación, género al que pertenece, idioma original y puntuación media.
- Un autor o autores, de los que sabemos el nombre, seudónimo y la nacionalidad.
- Los libros se organizan en líneas editoriales que tienen un nombre y una referencia, cada línea editorial se crea con el primer libro a la que pertenece.
- Los lectores pueden escribir críticas acerca de un libro.
- Las críticas tienen una puntuación de 1 a 10, fecha en la que se hizo, nombre de la persona, nacionalidad del que escribe la crítica, edad y el comentario realizado.

Este diagrama podría ser una posible solución.



Actividad diagramas de clases II

Seguro que aún tienes ganas de seguir practicando. Una vez que empiezas a hacer diagramas, ¡no puedes parar!

Se quiere realizar una aplicación para la gestión de pedidos de una empresa. Representa el diagrama de clases de dicha aplicación con la siguiente especificación:

- La empresa desea guardar la información de los pedidos realizados por los clientes indicando la fecha de realización y la localidad donde reside la empresa cliente.
- Cada pedido constará de varios productos.
- El precio total del pedido se calcula con todos los productos incluidos, indicando su cantidad e impuestos.
- El sistema debe saber el stock de cada producto.
- El pago de cada pedido se puede hacer de tres formas diferentes: con tarjeta de crédito, pidiendo la fecha de caducidad y número; en efectivo, pidiendo el tipo de moneda; y mediante cheque, indicando el nombre y el banco.

Este diagrama podría ser una posible solución.

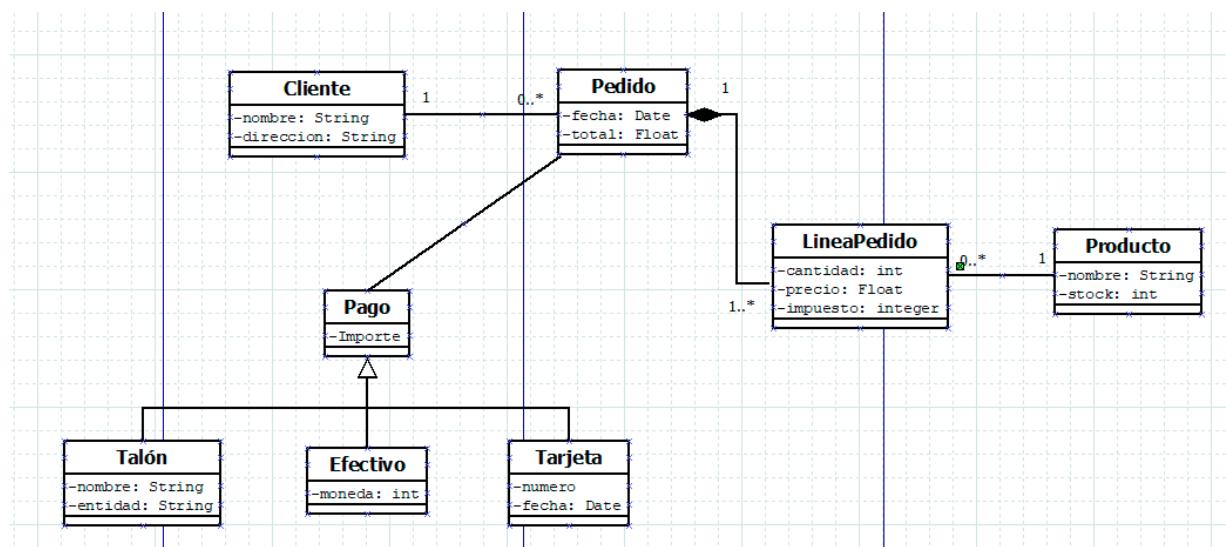


Diagrama de estados

Un diagrama de estados es un gráfico de estados y transiciones.

Describe cómo evoluciona un objeto a lo largo de su vida en función de los eventos que recibe. Además, representa los diferentes estados de un objeto y sus transiciones.

Los diagramas de estado se forman con tres tipos de elementos:

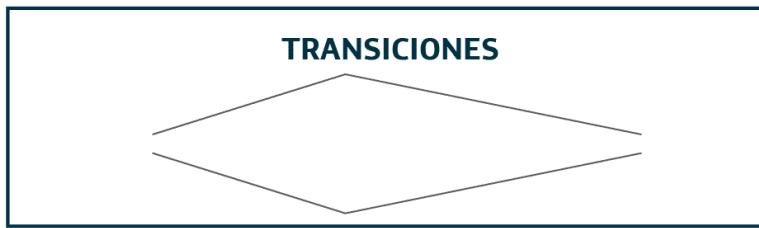
Estado

Condición o situación en la vida de un objeto durante la cual satisface alguna condición, realiza alguna actividad o espera algún evento.



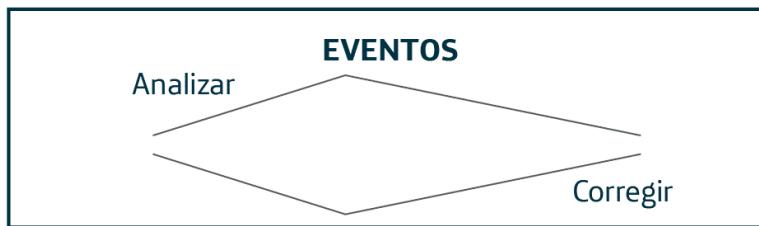
Transiciones

Relación entre dos estados que indica que un objeto que esté en el primer estado, realizará ciertas acciones y entrará en el segundo estado cuando ocurra un evento significativo y se satisfagan unas condiciones especificadas.



Eventos

Especificación de un acontecimiento significativo que ocupa un lugar en el tiempo y el espacio. Es la aparición de un estímulo que puede disparar una transición de estado.



Ejemplo

En el siguiente ejemplo podemos ver, mediante un diagrama de estados, el ciclo de vida de un objeto instanciado en una clase que representa una oferta de petición de un servicio efectuada por un cliente.

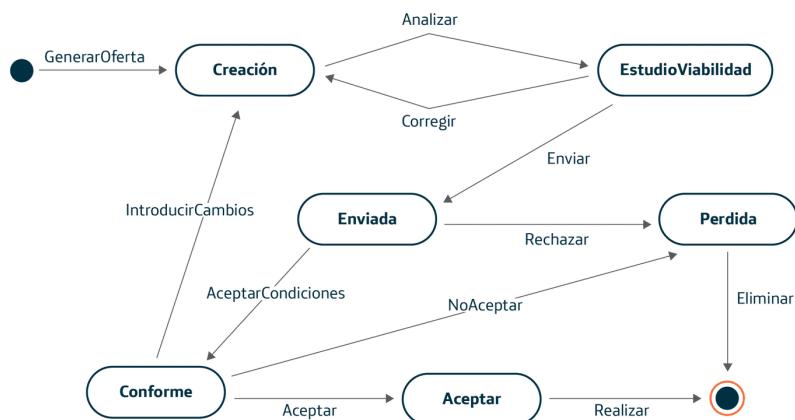
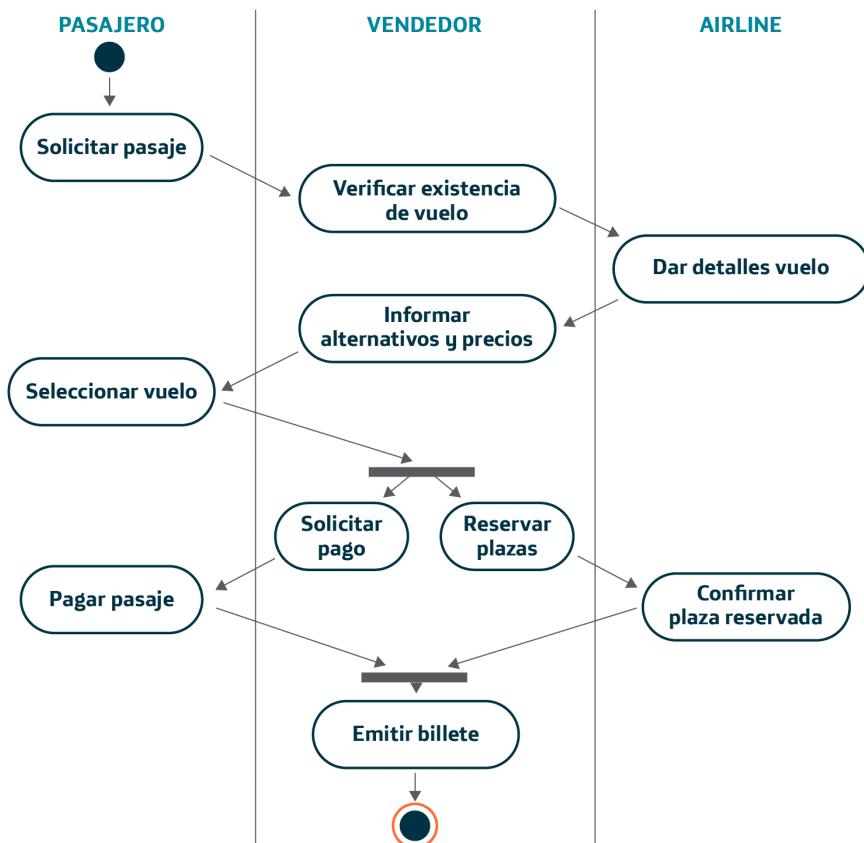


Diagrama de estados.

Diagrama de actividad

Los diagramas de actividad son una variante de los diagramas de estado, organizados respecto a las acciones y, principalmente, destinados a representar el comportamiento interno de un método o la realización de una operación o de un caso de uso.

Están compuestos de actividades unidas por arcos secuenciadas en el tiempo. Las actividades suceden en el tiempo (de arriba abajo) y concurrentemente (de derecha a izquierda).



Ejemplo de diagrama de actividad para el caso de uso de adquirir un billete de avión.

Diagramas de secuencia

Con los diagramas de secuencia mostramos los pasos que realiza una acción determinada, mostrando los diferentes pasos que intervienen en una acción.

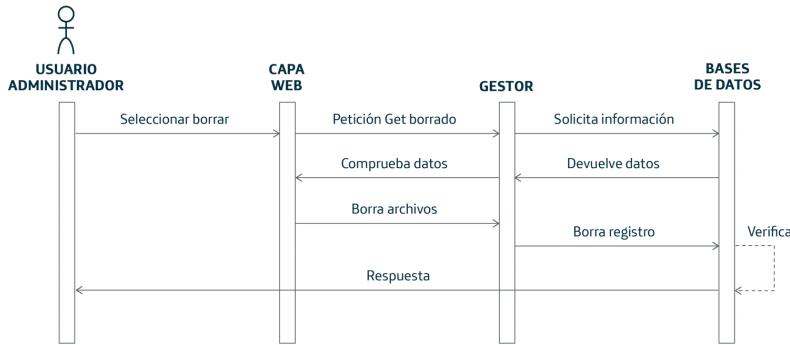


Diagrama de secuencia.

Los posibles usos que este tipo de diagramas permite son:

- 1 Mostrar la interacción de un conjunto de objetos en una aplicación a través del tiempo.
- 2 Dar detalle a los casos de uso, aclarándolos al nivel de mensajes de los objetos existentes.
- 3 Visualizar el uso de los mensajes de las clases diseñadas en el contexto de una operación. Cada línea del tiempo de un objeto a otro es una invocación a un método implementado en la clase del objeto.

Diagramas de colaboración

Un diagrama de colaboración es una forma de representar interacción entre objetos, alternativa al diagrama de secuencia.

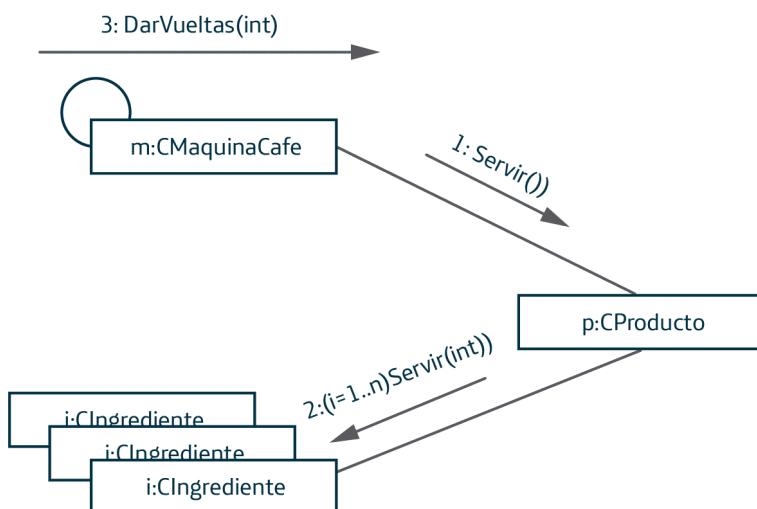
Los diagramas de colaboración representan de una vez:

1

El contexto de un grupo de objetos: objetos y enlaces.

2

La interacción entre estos objetos: envío de mensajes.



Ejemplo de diagrama de colaboración entre objetos.

Los dos tipos de diagramas de **secuencia** y de **colaboración** se pueden agrupar en el tipo de **diagramas de interacción**.

Ambos consisten en un conjunto de objetos y sus relaciones, incluyendo los mensajes que puedan ser enviados entre ellos.

Estos diagramas bien estructurados:

- Están enfocados en comunicar el aspecto dinámico de un sistema.
- Contienen solamente los elementos que son esenciales para entender un determinado aspecto o escenario.
- Proveen un detalle coherente con sus niveles de abstracción y deberían revelar solamente los adornos que son esenciales para su entendimiento. Dependiendo si se utiliza para documentar comportamientos de casos de uso en etapas de requisitos o análisis, o en tiempo de diseño para mostrar cómo interactúan objetos instanciados de la jerarquía de clases que resuelven una determinada funcionalidad, mostrarán más o menos detalle.

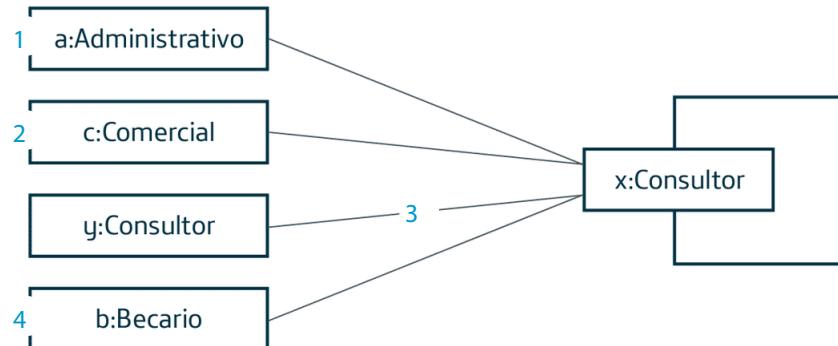
Diagramas de objetos

Los diagramas de objetos modelan las instancias de los elementos contenidos en los diagramas de clases. Están formados por objetos enlazados entre sí.

Muestran los objetos que en un instante determinado están colaborando para llevar a cabo una operación o funcionalidad.

Son parecidos a los diagramas de colaboración, pero sin mostrar los mensajes entre objetos.

En la imagen, cada objeto es una instancia de una clase y la clase *Consultor* tendrá que tener relaciones con las clases *Comercial*, *Administrativo*, *Becario* y con ella misma.



1 Objeto a

Es un objeto de la clase *Administrativo*.

2 Objeto c

Es un objeto de la clase *Comercial*.

3 Objetos x e y

Tanto *x* como *y* son objetos de la clase *Consultor*.

4 Objeto b

Es un objeto de la clase *Becario*.

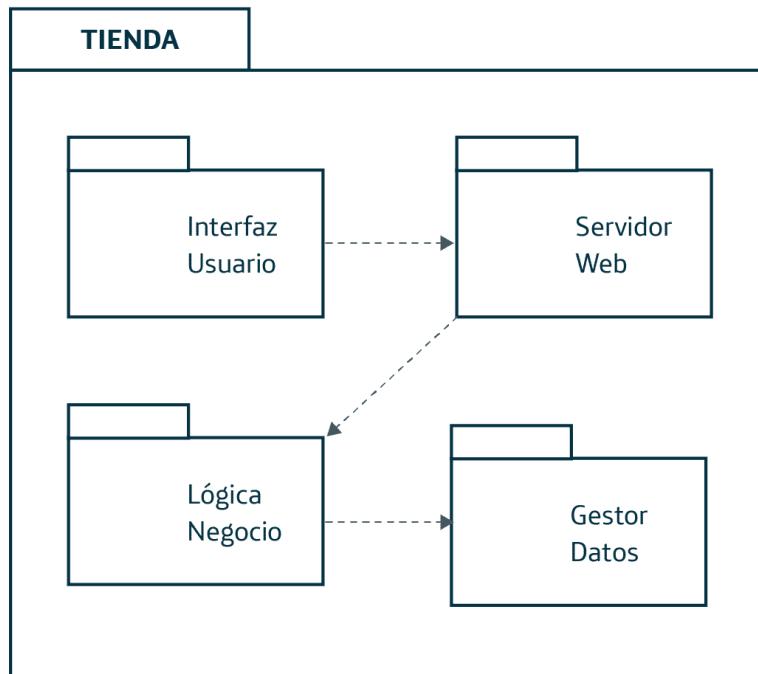
Es importante **diferenciar** el **diagrama de clases** que representa las relaciones entre los objetos y un **diagrama de objetos** que muestra las relaciones entre objetos **en un instante determinado**.

Diagrama de paquetes

En UML un paquete es un mecanismo de propósito general para organizar otros elementos en grupos. Los diagramas de paquetes están formados por paquetes entre los que existen unas relaciones de contención, de dependencia o de generalización.

Los paquetes se representan mediante un **rectángulo grande con un rectángulo más pequeño en su parte superior izquierda (ficha)**. Es el icono estándar de carpeta.

Si no se muestra el contenido entonces su nombre se sitúa dentro del rectángulo grande. Si se muestra el contenido entonces su nombre se sitúa dentro de la ficha.



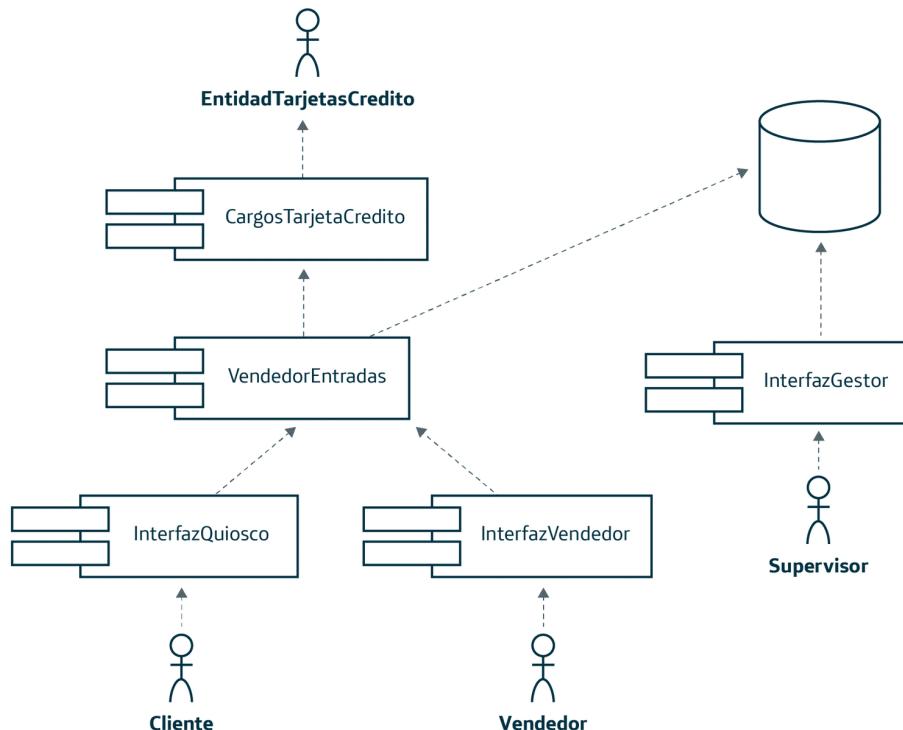
Ejemplo de un diagrama de paquetes.

Diagrama de componentes

Un componente de software es una parte física de un sistema. Están instalados, desplegados y configurados en los nodos de computación.

Los **diagramas de componentes** suelen mostrar las **relaciones de dependencia entre componentes**.

Un componente se representa como un rectángulo con dos rectángulos más pequeños que sobresalen a un lado. En estos diagramas se pueden incluir actores que interactúan con los componentes, sean "personas", dispositivos u otros sistemas.

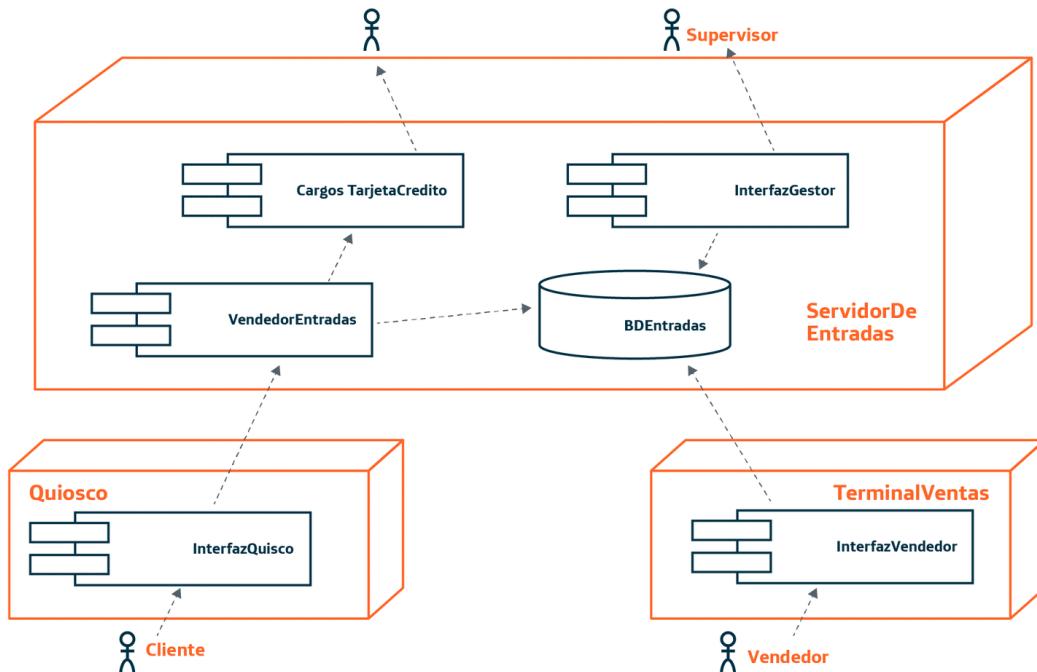


Ejemplo diagrama de componentes.

Diagramas de despliegue

Los diagramas de despliegue están formados por nodos relacionados entre ellos mediante líneas que representan una conexión.

Un **nodo** es un nombre genérico para todo tipo de recurso computacional con más o menos capacidad de computo y memoria.



Ejemplo de diagrama de despliegue con tres nodos con sus correspondientes actores.

Resumen

Has finalizado esta lección.

En esta unidad hemos visto los principales diagramas de UML para el modelado de un sistema, describiendo su utilidad y la correcta forma de crearlos.

Los modelos vistos son:

- **Diagramas de casos de uso.** Modelan los requisitos o interacciones con el sistema.
- **Diagramas de secuencia.** Modelan el paso de mensajes entre objetos en un intervalo de tiempo.
- **Diagramas de colaboración.** Modelan las interacciones entre los objetos en un escenario concreto.
- **Diagramas de estado.** Modelan los diferentes estados de un objeto en su tiempo de vida.
- **Diagramas de actividad.** Modelan el comportamiento de los casos de uso, operaciones u objetos.
- **Diagramas de clases.** Modelan la estructura estática de las clases.
- **Diagramas de objetos.** Modelan la estructura estática de los objetos.
- **Diagramas de paquetes.** Modelan la organización de los elementos del sistema en grupos.
- **Diagramas de componentes.** Modelan los componentes software que serán desplegados.
- **Diagramas de despliegue.** Modelan la disposición y arquitectura del sistema.



PROEDUCA