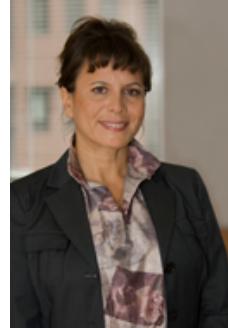


# Diseño orientado a objetos. Elaboración de diagramas de comportamiento.

## Caso práctico

En BK Programación continúan inmersos en el mundo de UML. A pesar de que han trabajado duro y han aprendido bastante acerca de este lenguaje de especificación, Ada se ha dado cuenta de que apenas han empezado a arañar la superficie de todas las posibilidades que les ofrece. De momento ya saben como crear un diagrama de clases bastante completo y como analizar un problema propuesto, sin embargo hay muchos aspectos del problema que no pueden modelar todavía, por ejemplo con solo el diagrama de clases no pueden saber qué se espera del sistema que van a construir, o en qué se deben basar para codificar los métodos, o simplemente, ¿Cómo colaboran los objetos de las clases que han creado para hacer alguna tarea que sea útil?



Ada decide que no pueden parar ahora, y que hay que hacer un esfuerzo final para que los conocimientos del equipo sean globales y puedan enfrentarse a cualquier desarrollo software con solvencia.

Al momento, Ada pone a su equipo manos a la obra.



[Ministerio de Educación y Formación Profesional.](#) (Dominio público)

**Materiales formativos de FP Online propiedad del Ministerio de Educación y Formación Profesional.**

[Aviso Legal](#)

# 1.- Introducción.

## Caso práctico

-Compañeros, creo que ahora no debemos conformarnos con modelar diagramas de clase y nada más, esto no nos da posibilidades de incluir ninguna información acerca del comportamiento de nuestro sistema. ¿Cómo especificamos la funcionalidad? O ¿qué acciones se realizan?, o ¿las restricciones? Necesitamos seguir estudiando diagramas que nos ayuden a especificar la dinámica del sistema. ¿Empezamos ahora mismo?



En el tema anterior vimos como crear un diagrama de clases para un problema determinado, esto nos ayuda a ver el problema con otra perspectiva y descubrir información nueva, sin embargo no tiene en cuenta elementos como la creación y destrucción de objetos, el paso de mensajes entre ellos y el orden en que deben hacerse, qué funcionalidad espera un usuario poder realizar, o como influyen elementos externos en nuestro sistema. Un diagrama de clases nos da información estática pero no dice nada acerca del comportamiento dinámico de los objetos que lo forman, para incluir éste tipo de información utilizamos los diagramas de comportamiento que incluyen:

- ✓ Diagramas de casos de uso.
- ✓ Diagramas de actividad.
- ✓ Diagramas de estados.
- ✓ Diagramas de interacción.
  - ◆ Diagramas de secuencia.
  - ◆ Diagramas de comunicación/colaboración.
  - ◆ Diagramas de interacción.
  - ◆ Diagramas de tiempo.

## Para saber más

En el siguiente enlace tienes una descripción y algunos ejemplos de todos los diagramas UML, puedes usarlo como complemento a lo que vamos a ver en la unidad. No obstante te animo a que busques en la web información y ejemplos diferentes que te ayuden a seguir los contenidos.

[Introducción a UML.](#)

## 2.- Diagramas de casos de uso.

### Caso práctico

-Empezaremos por el principio. Cuando estamos diseñando software es esencial saber cuales son los requerimientos del sistema que queremos construir, y necesitamos alguna herramienta que nos ayude a especificarlos de una manera clara, sistemática, y que nuestros clientes puedan entender fácilmente, ya que es imprescindible que nos pongamos de acuerdo en lo que realmente queremos hacer. ¿Alguna idea?



-¿No bastaría con hacer una lista de requerimientos y describirlos exhaustivamente?

-No, una descripción textual puede inducir a errores de interpretación y suele dejar cabos sueltos, y no contempla otra información, como quien realiza las acciones que describimos, por ejemplo. Necesitamos algo más específico.

Lo que Ada necesita son los diagramas de casos de uso.

Al construir software es esencial saber cuáles son los **requerimientos** del sistema que se desea crear, y se precisa alguna herramienta que ayude a especificarlos de una manera clara, sistemática y que los clientes puedan entender fácilmente.

Pero, ¿no bastaría con hacer una lista de requerimientos y describirlos exhaustivamente?. No, una descripción textual puede inducir a errores de interpretación y suele dejar cabos sueltos. La solución puede ser los **diagramas de casos de uso**.

Los **diagramas de casos** de uso son un elemento fundamental en la etapa de **análisis de un sistema** desde la perspectiva de la orientación a objetos porque resuelven uno de los principales problemas en los que se ve envuelto el proceso de producción de software: la falta de comunicación entre el equipo de desarrollo y el equipo que necesita de una solución software. Un diagrama de casos de uso nos ayuda a determinar **QUÉ** puede hacer cada tipo diferente de usuario con el sistema, en una forma que los no versados en el mundo de la informática o, más concretamente el desarrollo de software, pueda entender.

Los diagramas de casos de uso documentan el comportamiento de un sistema desde el punto de vista del usuario. Por lo tanto los casos de uso determinan los **requisitos funcionales del sistema**, es decir, representan las funciones que un sistema puede ejecutar.

Un diagrama de casos de uso es una visualización gráfica de los requisitos funcionales del sistema, que está formado por casos de uso (se representan como elipses) y los actores que interactúan con ellos (se representan como monigotes). Su principal **función** es dirigir el proceso de creación del software, definiendo qué se espera de él, y su **ventaja** principal es la facilidad para interpretarlos, lo que hace que sean especialmente útiles en la comunicación con el cliente.

### Reflexiona

Los diagramas de casos de uso se crean en las primera etapa de desarrollo del software, y se enmarcan en el proceso de análisis, para definir de forma detallada la funcionalidad que se

espera cumpla el software, y que, además, se pueda comunicar fácilmente al usuario, pero, ¿termina aquí su función?

[Mostrar retroalimentación](#)

En absoluto, de los diagramas de casos de uso se desprenden otros (normalmente se realiza antes que el diagrama de clases) que describen tanto la estructura del sistema como su comportamiento, lo que influye directamente en la implementación del sistema y en su arquitectura final. Por otra parte al describir específicamente qué se espera del software también se usa en la fase de prueba, para verificar que el sistema cumple con los requisitos funcionales, creándose muchos de los casos de prueba (pruebas de caja negra) directamente a partir de los casos de uso.

## **2.1.- Elementos del diagrama de casos de uso**

---

Los elementos de un diagrama de casos de uso son los siguientes:

- Actores.
- Casos de uso.
- Relaciones

## 2.1.1.- Actores.

### Caso práctico

-Como decía, uno de los principales problemas de una descripción textual es que no permite especificar adecuadamente qué personas o entidades externas interactúan con el sistema. Ahora tenemos una herramienta que tiene esto muy en cuenta.



Los **actores** representan un tipo de usuario del sistema. Se entiende como usuario cualquier cosa externa que interactúa con el sistema. No tiene por qué ser un ser humano, puede ser otro sistema informático o unidades organizativas o empresas.

Los **actores** representan los tipos de **usuario que interactúan con el sistema** (un ser humano, un PC, una empresa ...) . Es importante entender la diferencia entre actores y los usuarios, por ejemplo, un usuario puede interpretar diferentes roles según la operación que esté ejecutando, cada uno de estos roles representará un actor diferente. Por otro lado, cada actor puede ser interpretado por diferentes usuarios.

Por ejemplo, el dueño de una panadería podrá aparecer en un diagrama de casos de uso con los roles de administrador y de cocinero, a su vez, puede tener otro usuario contratado, de forma que el actor cocinero podrá ser "interpretado" tanto por el dueño como por el empleado.

#### Tipos de actores:

- **Primarios:** interaccionan con el sistema para explotar su funcionalidad. Trabajan directa y frecuentemente con el software.
- **Secundarios:** soporte del sistema para que los primarios puedan trabajar. Son precisos para alcanzar algún objetivo.
- **Iniciadores:** es posible que haya casos de uso que no sean iniciados por ningún usuario, en ese caso se podrá considerar un actor "tiempo" o "sistema" que asuma el arranque del caso.

Los actores se representan mediante la siguiente figura:



Es posible que haya casos de uso que no sean iniciados por ningún usuario, o algún otro elemento software, en ese caso se puede crear un actor "Tiempo" o "Sistema".

# Autoevaluación

¿Un sistema software externo, como una entidad para la autentificación de claves, podría considerarse como un actor en un diagrama de casos de uso?

- Verdadero.
- Falso.

Efectivamente, un actor no tiene porqué ser una persona física, es cualquier entidad que interaccione con el sistema.

No es así, un actor puede ser cualquier entidad externa que interactúe con el sistema, sea persona, empresa o software.

## Solución

1. Opción correcta
2. Incorrecto

## 2.1.2.- Casos de uso.

### Caso práctico

-Vale, pero lo que verdaderamente queremos es identificar la funcionalidad del sistema ¿no?, ¿cómo hace esta herramienta la descripción de la funcionalidad?



Se utilizan casos de uso para especificar tareas que deben poder llevarse a cabo con el apoyo del sistema que se está desarrollando.

Un **caso de uso** especifica una secuencia de acciones, incluyendo variantes, que el sistema puede llevar a cabo, y que producen un resultado observable de valor para un actor concreto.

El conjunto de casos de uso forma el "**comportamiento requerido**" de un sistema. El objetivo principal de elaborar un diagrama de casos de uso no es crear el diagrama en sí, sino la descripción que de cada caso se debe realizar, ya que esto es lo que ayuda al equipo de desarrollo a crear el sistema a posteriori. Junto al diagrama, por cada caso de uso se crea una tabla con una descripción textual, en la que se deben incluir, al menos, los siguientes datos (a los que se denomina **contrato**).

- ✓ **Nombre:** nombre del caso de uso.
- ✓ **Actores:** aquellos que interactúan con el sistema a través del caso de uso.
- ✓ **Propósito:** breve descripción de lo que se espera que haga.
- ✓ **Precondiciones:** aquellas que deben cumplirse para que pueda llevarse a cabo el caso de uso.
- ✓ **Flujo normal:** flujo normal de eventos que deben cumplirse para ejecutar el caso de uso exitosamente, desde el punto de vista del actor que participa y del sistema.
- ✓ **Flujo alternativo:** flujo de eventos que se llevan a cabo cuando se producen casos inesperados o poco frecuentes. No se deben incluir aquí errores como escribir un tipo de dato incorrecto o la omisión de un parámetro necesario.
- ✓ **Postcondiciones:** las que se cumplen una vez que se ha realizado el caso de uso.

**Casos de Uso**

User Case	Actor	Date	Brief Description	Pre-conditions	Post-conditions	Flow of Events	Actor Input	System Response
Usecase1	Actor1	29-agosto-2011 13:56:54						

La representación gráfica de un caso de uso se realiza mediante un óvalo o elipse, y su descripción se suele hacer rellenando una o más tablas como la de la imagen (obtenida de la herramienta Visual Paradigm).

### Autoevaluación

**"Tras comprobar todos los artículos el pedido queda en el almacén a la espera de ser recogido."**

**¿Dónde incluirías esta afirmación sobre un caso de uso en un contrato?**

- En el flujo de eventos normal.
- En el flujo de eventos alternativo.
- En las precondiciones.
- En las postcondiciones.

Falso, no es una acción atómica que pueda llevarse a cabo junto con otras para realizar la acción del caso de uso.

Incorrecto, no es una acción atómica que pueda llevarse a cabo junto con otras para realizar la acción del caso de uso en un caso particular del mismo.

No parece que sea una condición que deba cumplirse para poder ejecutar un caso de uso.

Correcto, es una condición que se cumple una vez que el caso de uso se ha ejecutado correctamente.

## Solución

1. Incorrecto
2. Incorrecto
3. Incorrecto
4. Opción correcta

## 2.1.3.- Relaciones.

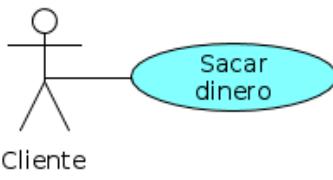
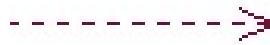
### Caso práctico

-De acuerdo, y ahora ¿Cómo asociamos a los actores con los casos de uso que pueden realizar?

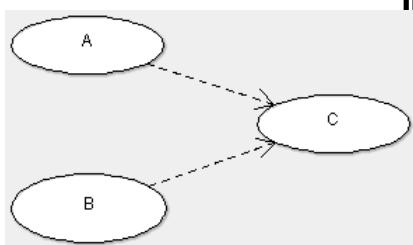


Los diagramas de casos de uso son ..... grafos no conexos en los que los nodos son **actores** y **casos de uso**, y las aristas son las **relaciones** que existen entre ellos. Las relaciones representan qué actores realizan las tareas descritas en los casos de uso, en concreto qué actores inician un caso de uso. Pero además existen otros tipos de relaciones que se utilizan para especificar relaciones más complejas, como uso o herencia entre casos de uso o actores.

Existen diferentes tipos de relaciones entre elementos:

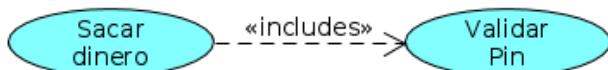
Relación	Descripción/ Ejemplo
<b>Asociación</b>	
	Representa la relación entre el actor y un caso de uso en el que participa.  <b>Ejemplo:</b> Relación entre el caso de uso <i>sacar dinero</i> y el <i>cliente</i> de un banco.
 A UML association diagram. On the left, there is a simple stick figure icon with the word "Cliente" written below it. A straight line connects this figure to a light blue oval on the right, which contains the text "Sacar dinero".	
<b>Inclusión (include - use)</b>	
 Include	Se trata de una relación entre casos de uso. La ejecución de un caso de uso implica <b>necesariamente</b> la ejecución del segundo.

Esta relación es muy útil cuando se desea especificar algún comportamiento común en dos o más casos de uso, aunque es frecuente cometer el error de utilizar esta técnica para hacer subdivisión de funciones, por lo que se debe tener mucho cuidado cuando se utilice.



#### Ejemplo 1:

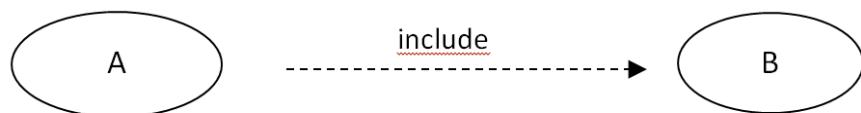
Al ejecutar el caso de uso *sacar dinero*, **obligatoriamente** se ejecuta el caso de uso *validar pin* de la tarjeta de crédito.



#### Ejemplo 2:

Por ejemplo, a la hora de hacer un pedido se debe buscar la información de los artículos para obtener el precio, es un proceso que necesariamente forma parte del caso de uso, sin embargo también forma parte de otros, como son el que visualiza el catálogo de productos y la búsqueda de un artículo concreto, y dado que tiene entidad por sí solo se separa del resto de casos de uso y se incluye en los otros tres.

En el siguiente gráfico se representa que A usa B, es decir, que **A siempre ejecuta B**.



#### Extensión (extend)

extend

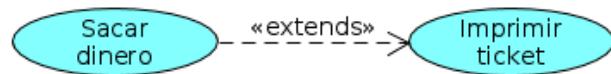
Se trata de una relación entre casos de uso. La ejecución de un caso de uso **puede** provocar la ejecución del segundo

Se utiliza una relación entre dos casos de uso de tipo "extends" cuando se desea especificar que el comportamiento de un caso de uso es diferente dependiendo de ciertas circunstancias.

La principal función de esta relación es simplificar el flujo de casos de uso complejos. Se utiliza cuando existe una parte del caso de uso que se ejecuta sólo en determinadas ocasiones, pero no es imprescindible para su completa ejecución. Cuando un caso de uso extendido se ejecuta, se indica en la especificación del caso de uso como un **punto de extensión**. Los puntos de extensión se pueden mostrar en el diagrama de casos de uso.

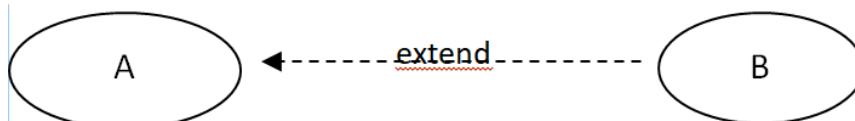
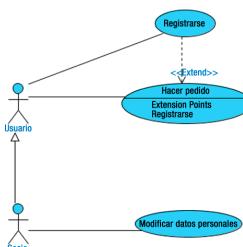
#### Ejemplo 1:

*Imprimir ticket* es consecuencia del caso de uso *sacar dinero*, pero su ejecución es opcional a que sea requerida por el cliente.



#### Ejemplo 2:

Cuando un usuario hace un pedido si no es socio se le ofrece la posibilidad de darse de alta en el sistema en ese momento, pero puede realizar el pedido aunque no lo sea.



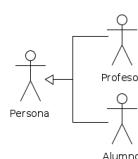
A opcionalmente ejecuta B.

### Generalización

Se utiliza para representar relaciones de herencia entre casos de uso o actores. No se contemplan generalizaciones combinadas entre actores y casos de uso.

Se utiliza cuando se tiene uno o más casos de uso que son especificaciones de un caso más general.

Por ejemplo, entre actores: tanto *profesor* como *alumno* son casos particulares del actor *persona*.



Ejemplos, entre casos de uso:

Un ejemplo de generalización de casos de uso sería la compra de artículos en un comercio, pudiendo considerarse la compra de alimentos o de bebidas. Ambos tipos de compras tendrán las características heredadas del caso de uso compra general, más las particulares definidas para cada caso.

# Autoevaluación

Supón el siguiente sistema que modela el caso de uso "Servir pedido" en el que el Empleado de almacén revisa si hay suficientes artículos para hacer el pedido y si todo es correcto, el pedido se embala y se envía:



¿Qué tipo de relación emplearías en el modelo del dibujo?

- Asociación.
- Generalización.
- extends.
- Include.

Incorrecto, no se establecen relaciones de asociación entre dos casos de uso, solo entre actores y casos de uso.

Falso, Consultar stock no se puede considerar un caso particular de Servir pedido ni lo amplía.

No es adecuado utilizar extends porque la comprobación del stock debe hacerse necesariamente, no en algún caso particular.

Correcto. Así, la consulta de existencias debe realizarse necesariamente y, además, tiene entidad suficiente como para ser un caso de uso por sí mismo, que puede usarse en otros casos.

## Solución

1. Incorrecto
2. Incorrecto
3. Incorrecto
4. Opción correcta

## 2.2.- Elaboración de casos de uso.

### Caso práctico

Después de analizar todos los elementos que formen un diagrama de casos de uso el equipo de Ada está preparado para hacer frente a su primer diagrama de casos de uso.

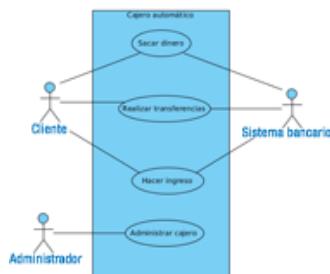


En los **diagramas de casos** se hace una **abstracción de la realidad** en la que representamos **qué cosas** pueden hacerse en nuestro sistema y **quién** las va a hacer.

Necesitamos diagramas cuya información permita al equipo de desarrollo la toma de decisiones adecuadas en la fase de análisis y diseño (cumpliendo los requisitos), así como que sean útiles en la fase de implementación en un lenguaje orientado a objetos.

Partiremos de una descripción lo más detallada posible del problema a resolver y trataremos de detectar aspectos como:

- Usuarios que interactúan con el sistema, para obtener los actores.
- Tareas que realizan estos actores para determinar los casos de uso más genéricos.
- Refinar el diagrama analizando los casos de uso más generales para detectar casos relacionados por inclusión, extensión y generalización.



## 2.3.- Escenarios.

### Caso práctico

Ada continua la investigación, junto con el equipo de BK programación, que una vez ha creado su primer diagrama de casos de uso, se da cuenta de que realmente es una herramienta muy útil a la hora de definir la funcionalidad de un sistema. Continuando con la investigación descubren una ventaja adicional, utilizando los flujos de eventos, pueden describir interacciones concretas de los actores con el sistema, estas interacciones son los escenarios.



Un caso de uso debe especificar un comportamiento deseado, pero no imponer cómo se llevará a cabo ese comportamiento, es decir, debe decir QUÉ pero no CÓMO. Esto se realiza utilizando escenarios que son casos particulares de un caso de uso.

Un **escenario** es una ejecución particular de un caso de uso que se describe como una secuencia de eventos. Un caso de uso es una generalización de un escenario.

Por ejemplo, para el caso de uso hacer pedido podemos establecer diferentes escenarios:

Un posible escenario podría ser:

Realizar pedido de unos zapatos y unas botas.

1. El usuario inicia el pedido.
2. Se crea el pedido en estado "en construcción".
3. Se selecciona un par de zapatos "Lucía" de piel negros, del número 39.
4. Se selecciona la cantidad 1.
5. Se recupera la información de los zapatos y se modifica la cantidad a pagar sumándole 45 €.
6. Se selecciona un par de botas "Aymara" de ante marrón del número 40.
7. Se selecciona la cantidad 1.
8. Se recupera la información de las botas y se modifica la cantidad a pagar sumándole 135 €.
9. El usuario acepta el pedido.
10. Se comprueba que el usuario es, efectivamente socio.
11. Se comprueban los datos bancarios, que son correctos.
12. Se calcula el total a pagar añadiendo los gastos de envío.
13. Se realiza el pago a través de una entidad externa.
14. Se genera un pedido para el usuario con los dos zapatos que ha comprado, con el estado "pendiente".

Los escenarios pueden y deben posteriormente documentarse mediante diagramas de secuencia.

## 2.4.- Ejercicio resuelto 1 ("ZAPATERÍA TACÓN DE ORO") (Elaboración de un diagrama de casos de uso).

### Ejercicio Propuesto

**Descripción del problema: "El tacón de oro".**

La zapatería Tacón de oro ha decidido crear un espacio web para ampliar su línea de negocio, así sus usuarios podrán adquirir los artículos: zapatos, bolsos y complementos que se venden en la tienda.

Los usuarios del sistema navegarán por la web para ver los artículos, zapatos, bolsos y complementos que se venden en la tienda. De los artículos nos interesa su nombre, descripción, material, color, precio y stock. De los zapatos nos interesa su número y el tipo. De los bolsos nos interesa su tipo (bandolera, mochila, fiesta). De los complementos (cinturones y guantes) su talla.

Los artículos se organizan por campañas para cada temporada (primavera/verano y otoño/ invierno) de cada año.

Los artículos son de fabricación propia, pero, opcionalmente, pueden venderse artículos de otras firmas. De las firmas nos interesa saber su nombre, CIF y domicilio fiscal. La venta de artículos de firma se realiza a través de proveedores, de forma que un proveedor puede llevar varios artículos de diferentes firmas, y una firma puede ser suministrada por más de un proveedor. Los artículos pertenecen a una firma solamente. De los proveedores debemos conocer su nombre, CIF, y domicilio fiscal.

Los usuarios pueden registrarse en el sitio web para hacerse socios. Cuando un usuario se hace socio debe proporcionar los siguientes datos: nombre completo, correo electrónico y dirección.

Los socios pueden hacer pedidos de los artículos. Los usuarios pueden consultar todos los productos que tienen a su disposición, pero para realizar compras han de registrarse como socios.

Para comprar productos, se generan pedidos. Un pedido está formado por un conjunto de detalles de pedido que son parejas formadas por artículo y la cantidad. De los pedidos interesa saber la fecha en la que se realizó y cuánto debe pagar el socio en total. El pago se hace a través de tarjeta bancaria, cuando se va a pagar una entidad bancaria comprueba la validez de la tarjeta. De la tarjeta interesa conocer el número.

Las campañas son gestionadas por el administrativo de la tienda que se encargará de dar de baja la campaña anterior y dar de alta la nueva siempre que no haya ningún pedido pendiente de cumplimentar.

Existe un empleado de almacén que revisa los pedidos diariamente y los cumplimenta. Esto consiste en recopilar los artículos que aparecen en el pedido y empaquetarlos. Cuando el paquete está listo se pasa al almacén a la espera de ser repartido. Del reparto se encarga una empresa de transportes que tiene varias rutas preestablecidas. Según el destino del paquete (la dirección del socio) se asigna a una u otra ruta. De la empresa de transportes se debe conocer su nombre, CIF y domicilio fiscal. Las rutas tienen un área de influencia que determina los destinos, y unos días de reparto asignados. Se debe conocer la fecha en la que se reparte el pedido. Si se produce alguna incidencia durante el reparto de algún pedido se almacena la fecha en la que se ha producido y una descripción.

Los socios pueden visualizar sus pedidos y una vez comprobados, puede cancelarlos (siempre y cuando no hayan sido cumplimentados por el empleado de almacén) o confirmar la

compra. Las compras deberán ser abonadas a través de una entidad bancaria. Así mismo los socios pueden modificar sus datos personales.

### Se pide:

- Diagrama de casos de uso. Identifica los actores y casos de uso, incluye relaciones de asociación, identifica generalizaciones (de actores y de casos de uso). Si consideras alguna relación tipo include o extend, justifica su uso.
- Del diagrama que hayas obtenido en el apartado anterior, agrupa todos los casos de uso que hayas considerado en el proceso de gestión de pedidos en uno sólo. Para este apartado, se entiende que el pedido ya ha sido dado de alta y que el proceso normal de un pedido es que termine siendo comprado. Desarrolla su notación escrita (tabla del caso de uso).

[Mostrar retroalimentación](#)

Los diagramas de casos de uso quedan encuadrados en la fase de análisis de los proyectos, se entienden como una puesta en común entre el cliente y el analista sobre como entender requisitos funcionales.

Al utilizarse UML, se entiende que ambos interlocutores conocen la notación propia de estos diagramas y las reglas sobre como combinar los diferentes símbolos, de forma que no hay ambigüedades que sí podrían darse mediante una descripción escrita.

Completar un diagrama de casos de uso supone versionar el diagrama tantas veces como sean necesarias hasta que la vista del proyecto que se presenta al cliente garantiza que el requisito funcional ha sido entendido.

En nuestros ejercicios, sólo hay una descripción que no permite ir refinando el diagrama mediante sucesivas consultas con el cliente, por lo tanto, las soluciones propuestas podrán ser diversas, todas ellas válidas siempre que reflejen razonablemente lo propuesto en el enunciado.

A la hora de valorar el ejercicio, lo que no es interpretable, es que el uso de la notación sea el correcto y que se muestre una variedad de los símbolos que conocemos para estos diagramas.

Con estas consideraciones, se propone la siguiente solución:

#### Actores que participan en el problema.

Zapatería (web), usuario (que podrá ser socio o no socio), productos/artículos (que podrán ser zapatos o complementos), y pedidos.

#### Generalizaciones.

Se observan dos posibles generalizaciones, que en el desarrollo del diagrama podrán ser expresadas en relación a actores o en relación a los casos de uso en los que participan. En la solución propuesta se ha decidido hacer una generalización de actores -- usuario socio o no socio -- y de casos de uso -- comprar zapatos o complementos (que podría haber sido una generalización de productos) --.

#### Relaciones extends.

De la ejecución del caso de uso visualizar pedido se pueden derivar los casos de uso cancelar pedido y comprar artículo. Se trata de casos de uso que no tienen porque siempre llevarse a cabo.

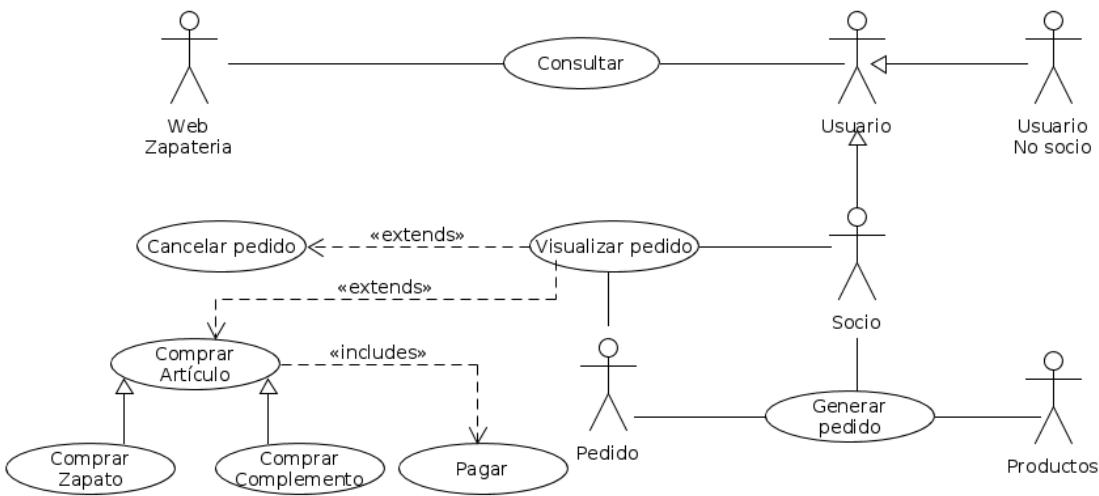
#### Relaciones includes/use.

Cuando se realiza la compra de un artículo, siempre habrá que pagar.

#### Notas:

- Puesto que todos los usuarios pueden consultar la web de la zapatería, la relación consultar web la hace el actor usuario (padre de la relación de generalización), en cambio, generar pedido sólo puede llevarse a cabo por los socios, así que es éste el que participa en este caso de uso.
- También podría haberse considerado el actor entidad bancaria, partícipe del caso de uso pagar.

Quedando el siguiente diagrama.



Como se indicaba en los contenidos del apartado 2.2, lo más importante en la elaboración de un diagrama de casos de uso, no es el diagrama en sí, sino la documentación de los casos de uso que es lo que permitirá desarrollar otros diagramas que ayuden en la codificación del sistema, y la elaboración de los casos de prueba de caja negra.

A modo de ejemplo vamos a desarrollar la documentación del caso de uso **Generar Pedido**, ya que, por su complejidad abarca todos los apartados que hemos visto. El ejemplo se hará con la herramienta Visual Paradigm for UML, aunque tu puedes usar la herramienta que consideres más oportuna

Los datos que debemos incluir para elaborar la documentación del caso de uso eran:

- ✓ **Nombre:** nombre del caso de uso.
- ✓ **Actores:** aquellos que interactúan con el sistema a través del caso de uso.
- ✓ **Propósito:** breve descripción de lo que se espera que haga.
- ✓ **Precondiciones:** aquellas que deben cumplirse para que pueda llevarse a cabo el caso de uso.
- ✓ **Flujo normal:** flujo normal de eventos que deben cumplirse para ejecutar el caso de uso exitosamente.
- ✓ **Flujo alternativo:** flujo de eventos que se llevan a cabo cuando se producen casos inesperados o poco frecuentes. No se deben incluir aquí errores como escribir un tipo de dato incorrecto o la omisión de un parámetro necesario.
- ✓ **Postcondiciones:** las que se cumplen una vez que se ha realizado el caso de uso.

Para incluir el nombre, actores, propósito, precondiciones y postcondiciones abrimos la especificación del caso de uso. Esto da lugar a la aparición de una ventana con un conjunto de pestañas que podemos llenar:

- ✓ En la pestaña "**General**" rellenamos el nombre "**Hacer pedido**", y tenemos un espacio para escribir una breve descripción del caso de uso, por ejemplo:
 

*"El cliente visualiza los productos que están a la venta, que se pueden seleccionar para añadirlos al pedido. Puede añadir tantos artículos como desee, cada artículo añadido modifica el total a pagar según su precio y la cantidad seleccionada.*

*Cuando el cliente ha llenado todos los productos que quiere comprar debe formalizar el pedido.*

*En caso de que el cliente no sea socio de la empresa antes de formalizar la compra se le indica que puede hacerse socio, si el cliente acepta se abre el formulario de alta, en caso contrario se cancela el pedido.*

*En caso de que se produzca algún problema con los datos bancarios se ofrecerá la posibilidad de volver a introducirlos.*

*Al finalizar un pedido se añade al sistema con el estado pendiente."*
- ✓ En la pestaña "**Valores etiquetados**" encontramos un conjunto de campos predefinidos, entre los que se encuentran el autor, precondiciones y postcondiciones, que podemos llenar de la siguiente manera:
 

**Autor:** usuario.

**Precondiciones:** Existe una campaña abierta con productos de la temporada actual a la venta.

**Postcondiciones:** Se ha añadido un pedido con un conjunto de productos para servir con el estado "pendiente" que deberá ser revisado.

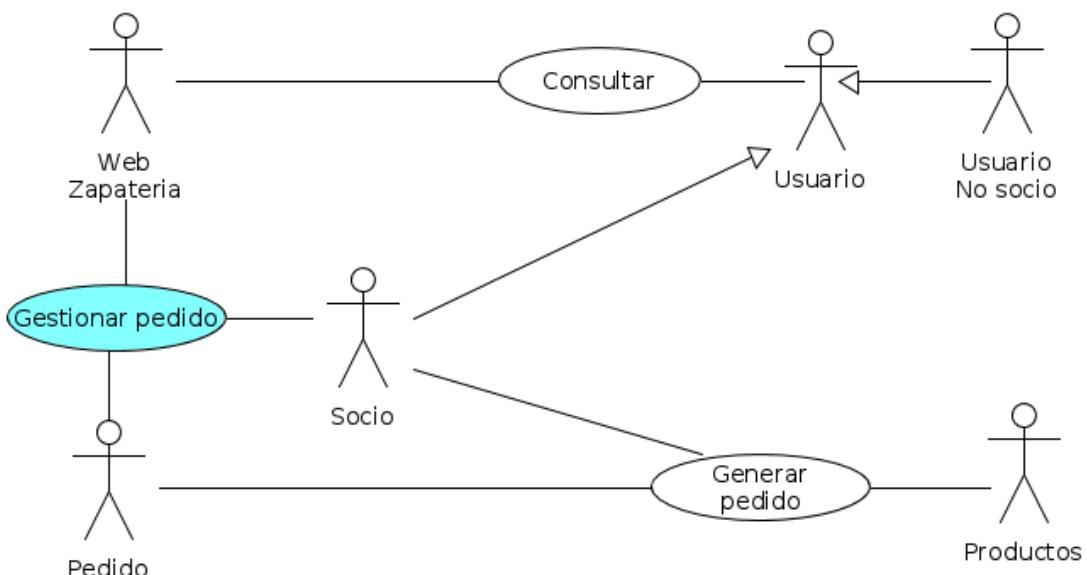
También se pueden incluir otros datos como la complejidad, el estado de realización del caso de uso la complejidad que no hemos visto en esta unidad.

Para incluir el resto de los datos en el caso de uso hacemos clic en la opción "**Open Use Case Details...**" del menú contextual, lo que da lugar a la aparición de una ventana con una serie de pestañas. En ellas se recupera la información de la especificación que hemos rellenado antes, además, podemos llenar el flujo de eventos del caso de uso, en condiciones normales usaríamos la pestaña "Flow of events", sin embargo como esta opción solo está disponible en la versión profesional, utilizaremos la pestaña "**Descripción**", que está disponible en la versión community. Para activarla pulsamos el botón "**Create/Open Description**".

Podemos añadir varias descripciones de diferentes tipos, pulsando el botón "**Nuevo**". Para añadir filas al flujo de eventos pinchamos en el botón. En principio añadimos la descripción principal, luego añadiremos otras alternativas.

#### **Tabla gestionar pedido.**

Una vez adaptado el diagrama para obtener el caso de uso gestionar pedidos, podría quedar como sigue:



La tabla resultante sería:

<b>Caso de uso</b>	Gestionar pedidos.		
<b>Actores</b>	Principal: usuario-socio. Secundarios: pedido, zapatería.		
<b>Propósito</b>	Gestión de los pedidos ya dados de alta en el sistema.		
<b>Pre-condiciones</b>	El pedido ha sido generado y está en modo activo. El usuario debe ser socio.		
<b>Flujo normal</b>	El flujo normal finaliza con la compra del pedido.	Visualizado del pedido.	Compra del pedido.
		Pago del pedido.	Fin de flujo.
<b>Flujos alternativos</b>	Cancelación del pedido.	Visualizado del pedido.	Cancelación del pedido. El pedido desaparece de la lista de pedidos activos.

		Fin de flujo.
Imposibilidad de efectuar el pago.	de	Visualizado del pedido.
		Compra del pedido.
		Error en la operación de pago. La compra del pedido no se completa.
		Fin de flujo.
<b>Post-condiciones</b>	Las post-condiciones son estados en los que ha de quedar el sistema siempre, sea cual sea el flujo ejecutado (normal o alternativo). Para este enunciado no se identifica ninguna post-condición en particular.	
<b>Requerimientos trazados</b>	Los que correspondan en el documento de requisitos. Recuerda que el tipo de requisitos que se consideran en los diagramas de casos de uso son siempre de tipo funcional.	
<b>Puntos de inclusión</b>	Ninguno. Este diagrama no dispone de otros casos de uso con los que establecer estas relaciones.	
<b>Puntos extensión</b>	Ninguno. Este diagrama no dispone de otros casos de uso con los que establecer estas relaciones.	

Esta es la descripción principal, en ella se describe el flujo normal de eventos que se producen cuando se ejecuta el caso de uso sin ningún problema.

### Flujo de eventos normal para el caso de uso Hacer Pedido.

<b>Use Case</b>	Hacer pedido	
<b>Author</b>	usuario	
<b>Date</b>	26-agosto-2011 13:56:56	
<b>Brief Description</b>	EL usuario selecciona un conjunto de artículos, junto con la cantidad de los mismos, para crear el pedido. Cuando se formaliza se comprueba que el usuario sea socio. A continuación se comprueban los datos bancarios, se realiza el cobro y se crea el pedido.	
<b>Preconditions</b>	Existe un catálogo de productos disponibles para pedir. El usuario está registrado. Los datos bancarios son correctos.	
<b>Post-conditions</b>	Se crea un pedido con los datos del usuario que lo realiza y los artículos solicitados.	
<b>Flow of Events</b>	<b>Actor Input</b>	<b>System Response</b>
1	Inicia el pedido.	
2		Se crea un pedido en estado "en construcción".
3	Selecciona un artículo.	
4	Selecciona una cantidad.	
5		Recupera la información del artículo para obtener el precio y modifica el

		precio total del pedido.
<b>6</b>	El proceso se repite hasta completar la lista de artículos.	
<b>7</b>	Se acepta el pedido.	
<b>8</b>		Se comprueba si el usuario es socio, si no lo es se le muestra un aviso para que se registre en el sitio.
<b>9</b>		Se comprueban los datos bancarios con una entidad externa.
<b>10</b>		Se genera: calcula el total, sumando los gastos de envío.
<b>11</b>		Se realiza el pago a través de la entidad externa.
<b>12</b>		Se almacena la información del pedido con el estado "pendiente".

Añadimos un par de descripciones alternativas para indicar que hacer cuando el usuario no es socio y cuando los datos bancarios no son correctos:

### **Flujo alternativo para el caso de uso Hacer Pedido cuando el usuario no está registrado.**

<b>Author</b>	usuario																			
<b>Date</b>	26-agosto-2011 17:56:35																			
<b>Brief Description</b>	Cuando el usuario hace un pedido si no está registrado se abre la opción de registro para que se dé de alta.																			
<b>Preconditions</b>	El usuario no está registrado. Existe un catálogo de artículos para hacer pedido. Los datos bancarios son correctos.																			
<b>Post-conditions</b>	El usuario queda registrado. Se crea un pedido con los datos del usuario que lo realiza y los artículos solicitados.																			
<b>Flow of Events</b>	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th></th> <th style="text-align: center;"><b>Actor Input</b></th> <th style="text-align: center;"><b>System Response</b></th> </tr> </thead> <tbody> <tr> <td><b>1</b></td><td>Inicia el pedido.</td><td></td></tr> <tr> <td><b>2</b></td><td></td><td>Se crea un pedido en estado "en construcción".</td></tr> <tr> <td><b>3</b></td><td>Selecciona un artículo.</td><td></td></tr> <tr> <td><b>4</b></td><td>Selecciona la cantidad.</td><td></td></tr> <tr> <td><b>5</b></td><td></td><td>Recupera la información del artículo para obtener su precio y modifica el precio total a pagar por el pedido.</td></tr> </tbody> </table>		<b>Actor Input</b>	<b>System Response</b>	<b>1</b>	Inicia el pedido.		<b>2</b>		Se crea un pedido en estado "en construcción".	<b>3</b>	Selecciona un artículo.		<b>4</b>	Selecciona la cantidad.		<b>5</b>		Recupera la información del artículo para obtener su precio y modifica el precio total a pagar por el pedido.	
	<b>Actor Input</b>	<b>System Response</b>																		
<b>1</b>	Inicia el pedido.																			
<b>2</b>		Se crea un pedido en estado "en construcción".																		
<b>3</b>	Selecciona un artículo.																			
<b>4</b>	Selecciona la cantidad.																			
<b>5</b>		Recupera la información del artículo para obtener su precio y modifica el precio total a pagar por el pedido.																		

<b>6</b>		Añade la información al pedido en creación.
<b>7</b>	EL proceso se repite hasta completar la lista de artículos del pedido.	
<b>8</b>	Acepta el pedido.	
<b>9</b>		Se comprueba si el usuario es socio.
<b>10</b>		Se invoca el registro de usuario.
<b>11</b>		Se comprueban los datos bancarios con una entidad externa.
<b>12</b>		Se genera: calcula el total, sumando los gastos de envío.
<b>13</b>		Se realiza el pago a través de la entidad externa.
<b>14</b>		El pedido queda almacenado con el estado "pendiente".

### Flujo alternativo para hacer pedido cuando los datos bancarios no son correctos.

<b>Author</b>	usuario	
<b>Date</b>	26-agosto-2011 18:14:35	
<b>Brief Description</b>	Una vez que se han seleccionado los artículos y se han introducido los datos del socio, al hacer la comprobación de los datos bancarios con la entidad externa se produce algún error, se da la posibilidad al usuario de modificar los datos o de cancelar el pedido.	
<b>Preconditions</b>	Existe un catálogo de productos disponibles para pedir. El usuario está registrado. Los datos bancarios no son correctos.	
<b>Post-conditions</b>	Se crea un pedido con los datos del usuario que lo realiza y los artículos solicitados.	
<b>Flow of Events</b>	Actor Input	System Response
<b>1</b>	Inicia el pedido.	
<b>2</b>		Se crea un pedido en estado "en construcción".
<b>3</b>	Selecciona un artículo.	
<b>4</b>	Selecciona una cantidad.	
<b>5</b>		Recupera la información del artículo para obtener el precio y modifica el precio total del pedido.

<b>6</b>	El proceso se repite hasta completar la lista de artículos.	
<b>7</b>	Se acepta el pedido.	
<b>8</b>	Acepta el pedido.	
<b>9</b>		Se comprueban los datos bancarios con una entidad externa, fallando la comprobación.
<b>10</b>		Se solicitan los datos de nuevo.
<b>11</b>	Introduce los datos de nuevo.	
<b>12</b>		Se repite el proceso hasta que se acepten los datos bancarios o se cancele la operación.
<b>13</b>		Se genera: calcula el total, sumando los gastos de envío.
<b>14</b>		Se realiza el pago a través de la entidad externa.
<b>15</b>		Se almacena la información del pedido con el estado "pendiente".

El resto de casos se uso se documentan de forma similar hasta completar la descripción formal de la funcionalidad del sistema.

## Debes conocer

La elaboración de casos de uso no es un proceso inmediato, en la siguiente presentación tienes la descripción de como elaborar el diagrama de casos de uso del sistema con el que estamos trabajando.

Revísalo con cuidado para comprender los conceptos que acabamos de ver.

## Primeros pasos

- Antes de elaborar el diagrama tienes que leer con detenimiento el documento con la especificación del problema a resolver y asegurarte de que entiendes la idea central del problema, crear una tienda virtual en la que se puedan realizar pedidos de los productos a la venta (zapatos). El proceso se centra en el pedido, desde poner a disposición del cliente los artículos en venta, pasando por la selección de artículos a pedir, la cumplimentación de toda la información necesaria para el pedido, pago, confección del pedido, envío y reajuste del stock en almacén, todo ello, a través de la web.



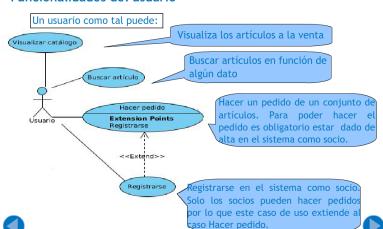
## Identificar funcionalidades

- Para facilitar la creación del diagrama vamos a ir sacando funcionalidades para cada usuario.
- Debemos recordar que una caso de uso representa una interacción de un actor con el sistema, que está relacionado con los requisitos funcionales de la aplicación final y que, en definitiva representa tareas que llevará a cabo el sistema.

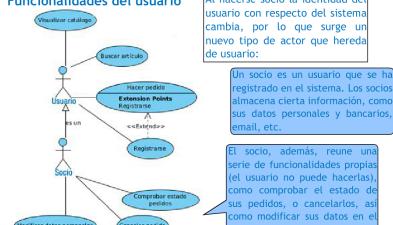
## Funcionalidades del usuario

- Cuando una persona se conecta al sistema lo primero que podrá hacer será visualizar el catálogo de la temporada.
- También puede hacer un pedido con uno o varios artículos del catálogo, para ello visualizará los artículos de forma que pueda seleccionar algunos de ellos e indicar la cantidad que quiere comprar.
- También puede hacer búsquedas por datos concretos de artículos.
- Cualquier persona que acceda al sistema puede darse de alta para ser socio.
- Así mismo, si es socio, podría comprobar el estado de sus pedidos y cancelarlos.

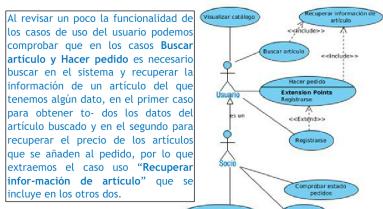
### Funcionalidades del usuario



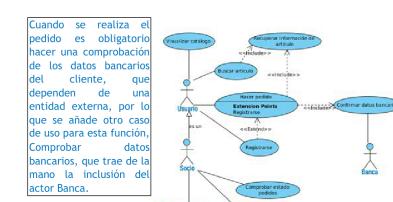
### Funcionalidades del usuario



### Funcionalidades del usuario



### Funcionalidades del usuario



### Funcionalidades del administrador

- El objetivo del administrador web es gestionar los contenidos de la web, en concreto de las diferentes campañas, ya que cada temporada se debe cerrar la campaña antigua, retirando los artículos de la temporada anterior y abrir la temporada nueva, añadiendo sus artículos. Para que se pueda cerrar una temporada es necesario que en el almacén se hayan gestionado todos sus pedidos, por lo que es obligatorio comprobarlo, antes de cerrar.

### Funcionalidades del administrador

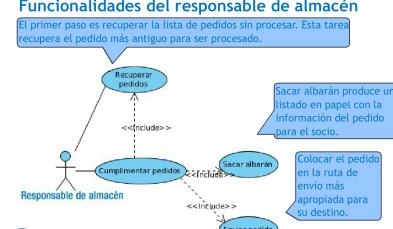


### Funcionalidades del responsable de almacén

- Es el encargado de leer los pedidos de los usuarios y cumplimentarlos, está será su única función, si bien, es una función complicada, ya que implica realizar una serie de tareas:

  - Seleccionar el pedido más antiguo.
  - Buscar los artículos a servir.
  - Empaquetarlos junto con un albarán para el socio.
  - Colocarlos en su ruta de envío.

### Funcionalidades del responsable de almacén



## 2.5.- Ejercicio resuelto 2 ("QUIJOTE")(Elaboración de un diagrama de casos de uso).

### Ejercicio Propuesto

La empresa Quijote se dedica a la venta de material informático puerta a puerta, ofrece sus productos a los clientes en sus propios domicilios. Sus empleados se organizan en dos grandes grupos: vendedores y publicitarios. Los publicitarios tratan de facilitar el acceso de los vendedores a los clientes para que éstos les hagan llegar los catálogos de productos y realicen las operaciones de ventas.

Los publicitarios anualmente encargan a la consultora Sancho un estudio de sus resultados, y en función de los datos que desprenda pueden realizar un análisis de mercado. Gracias a la información obtenida en el análisis se hacen campañas publicitarias en radio y televisión.

La política de la empresa Quijote se basa en tener grandes profesionales en sus filas, por lo que todos sus empleados reciben formación periódicamente.

#### Se pide:

- Diagrama de casos de uso. Identifica los actores y casos de uso, incluye relaciones de asociación, identifica generalizaciones. Si consideras alguna relación tipo include o extend, justifica su uso.
- Si agrupamos todos los casos de uso que hayas considerado en el proceso relacionado con los empleados de marketing/publicitarios en uno sólo. Desarrolla la notación escrita del caso de uso.

[Mostrar retroalimentación](#)

Los diagramas de casos quedan encuadrados en la fase de análisis de los proyectos, se entienden como una puesta en común entre el cliente y el analista sobre como entender requisitos funcionales.

Al utilizarse UML, se entiende que ambos interlocutores conocen la notación propia de estos diagramas y las reglas sobre como combinar los diferentes símbolos, de forma que no hay ambigüedades que sí podrían darse mediante una descripción escrita.

Completar un diagrama de casos de uso supone versionar el diagrama tantas veces como sean necesarias hasta que la vista del proyecto que se presenta al cliente garantiza que el requisito funcional ha sido entendido.

En nuestros ejercicios, sólo hay una descripción que no permite ir refinando el diagrama mediante sucesivas consultas con el cliente, por lo tanto, las soluciones propuestas podrán ser diversas, todas ellas válidas siempre que reflejen razonablemente lo propuesto en el enunciado.

A la hora de valorar el ejercicio, lo que no es interpretable, es que el uso de la notación sea el correcto y que se muestre una variedad de los símbolos que conocemos para estos diagramas.

Con estas consideraciones, se propone la siguiente solución:

#### Actores que participan en el problema.

Empleado (que podrá ser ventas o publicitario), cliente y consultora.

#### Generalizaciones.

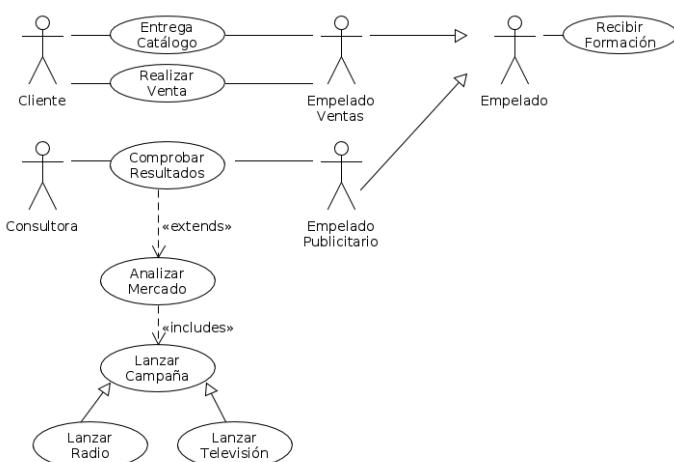
Se observan dos posibles generalizaciones, que en el desarrollo del diagrama podrán ser expresadas en relación a actores o en relación a los casos de uso en los que participan. En la solución propuesta se ha decidido hacer una generalización de actores -- empleado de ventas y empleado publicitario -- y de casos de uso -- lanzar campaña por radio o por televisión --.

#### Relaciones extends.

De la ejecución del caso de uso comprobar resultados se puede derivar el caso de uso analizar mercado.

**Relaciones includes/use.** Cuando se realiza un análisis de mercado, siempre se lanza una campaña publicitaria.

Quedando el siguiente diagrama.



#### Tabla promocionar Quijote.

Una vez adaptado el diagrama para obtener el caso de uso promocionar Quijote, podría quedar como sigue:



La tabla resultante sería:

Caso de uso	Promocionar Quijote.	
Actores	Empleado marketing (principal), consultor (secundario).	
Propósito	Promocionar la empresa para ganar nuevos clientes.	
Pre-condiciones	Seleccionar consultora con experiencia en el sector.	
Flujo normal	<p>Como flujo normal vamos a suponer que del estudio se genera un análisis de mercado, sería igualmente válida la decisión contraria.</p>	<p>El empleado solicita comprobar los resultados del año en curso a la consultora.</p> <p>La consultora concluye que se precisa un análisis de mercado de acuerdo al estado actual de la empresa.</p> <p>Se lanza una campaña publicitaria en radio.</p> <p>Se lanza una campaña publicitaria en televisión.</p> <p>Fin de flujo.</p>
Flujos alternativos	<p>El análisis de mercado no muestra potenciales clientes.</p>	<p>El empleado solicita comprobar los resultados del año en curso a la consultora.</p> <p>La consultora concluye que se precisa un análisis de mercado de</p>

		acuerdo al estado actual de la empresa, pero no se detectan clientes futuros.
		Fin de flujo.
	El presupuesto para las campañas publicitarias disponible no es suficiente	El empleado solicita comprobar los resultados del año en curso a la consultora.
		La consultora concluye que se precisa un análisis de mercado de acuerdo al estado actual de la empresa.
		Se lanza una campaña publicitaria en radio.
		Se suspende la campaña publicitaria en televisión por falta de fondos.
		Fin de flujo.
<b>Post-condiciones</b>	No se identifican en este ejercicio. Serían aquellas condiciones que se deben cumplir <u>siempre</u> al finalizar el caso de uso sea cual sea el flujo ejecutado.	
<b>Requisitos trazados</b>	Los que correspondan en el documento de requisitos. Recuerda que el tipo de requisitos que se consideran en los diagramas de casos de uso son siempre de tipo funcional.	
<b>Puntos de inclusión</b>	Ninguno. Este diagrama no dispone de otros casos de uso con los que establecer estas relaciones.	
<b>Puntos de extensión</b>	Ninguno. Este diagrama no dispone de otros casos de uso con los que establecer estas relaciones	

## 2.6.- Ejercicio resuelto 3 ("ALQUILER DE PISOS Y LOCALES") (Elaboración de un diagrama de casos de usos).

### Ejercicio Propuesto

Una empresa de alquiler de pisos y locales desea diseñar un sistema que cumpla los siguientes requisitos:

- Los propietarios previa identificación en el sistema, podrán dar de alta o baja un piso o un local. También podrán modificar los datos de ese piso o local.
- Los futuros inquilinos también deben identificarse antes de poder usar el sistema. Al acceder se les presenta un menú donde pueden elegir la acción a realizar:
  1. listar pisos y locales disponibles.
  2. solicitar el alquiler de un piso o local.

Para alquilar un local se le pedirá su email y para alquilar un piso su número de teléfono

#### Diagrama de casos de uso

1. Identificar los actores.
2. Identificar los casos de uso.
3. Implementar con UMLet el diagrama de casos de uso.

[Mostrar retroalimentación](#)

#### 1.-Identificar los actores.

Los actores identificados son : Propietario e Inquilino.

PropietarioEl propietario será el responsable de la gestión de los pisos o locales mediante el alta, la baja y las modificaciones de los mismos. Una vez identificado en el sistema podrá gestionar pisos o locales.

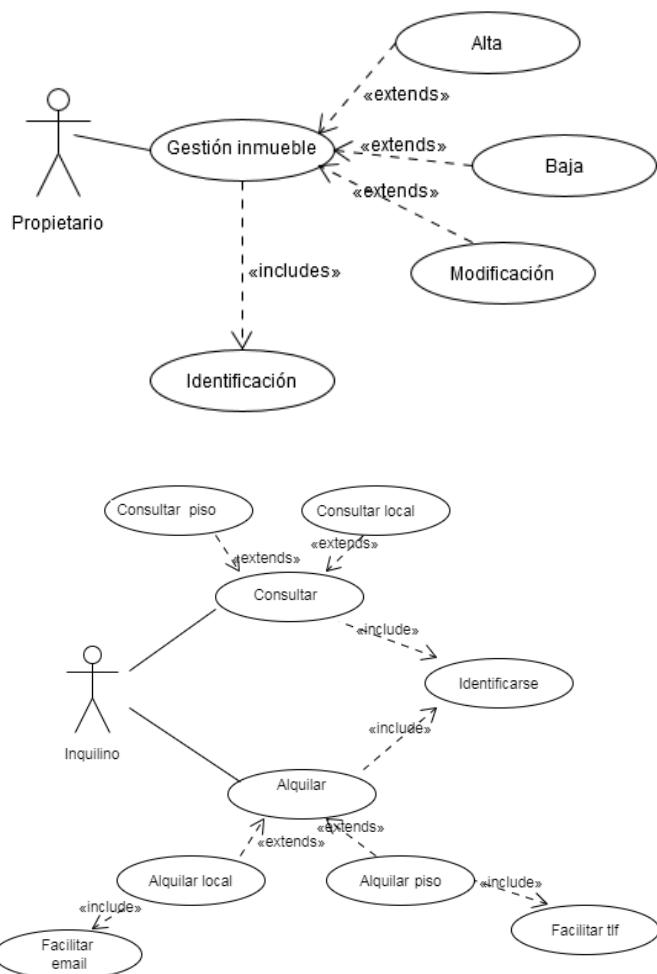
InquilinoEl inquilino consultará los pisos o locales y también podrá solicitar el alquiler. Debe identificarse en el sistema.

#### 2.-Identificar los casos de uso.

El propietario podrá dar de alta, baja dar de baja o modificar un inmueble. Si se ofrece un menú podemos usar extends con estos casos de uso y uno llamado gestión inmueble. El inquilino siempre que quiera gestionar un inmueble se identifica.

El inquilino podrá consultar o alquilar. En base a lo que elija podrá ser local o piso. Cuando alquile deberá facilitar teléfono o email según proceda. También debe identificarse para realizar las acciones.

#### 3.-Implementar con UMLet el diagrama de casos de uso.



## 3.-Diagrama de interacción

---

Una vez conocidos los diagramas de casos de uso, se hace necesario buscar la forma de representar como circula la información, los objetos que participan en los casos de uso, los mensajes que envían, y en el momento en que se producen. Disponer de esta información ayudará con posterioridad en el desarrollo de los diagramas de clases.

Los **diagramas de interacción** son vistas del sistema que muestran como grupos de objetos interactúan para un cierto comportamiento. Captan la ejecución de los casos de uso, representando a los actores que participan y los mensajes que se pasan.

Hay dos tipos de diagramas de interacción: **diagramas de secuencia** y **diagramas de colaboración**.

El diagrama de colaboración contiene la misma información que un diagrama de secuencia, pero la anotación es diferente.

## 3.1.- Diagramas de secuencia.

### Caso práctico

María se ha dado cuenta de que los casos de uso permiten, de una manera sencilla, añadir información sobre qué hace el sistema, sin embargo por completa que se la descripción de la secuencia de eventos no permite incluir información útil, como los objetos que intervienen en las tareas, y como se comunican.

-Tendríamos que buscar la forma de representar como circula la información, que objetos participan en los casos de uso, qué mensajes envían, y en qué momento, esto nos ayudaría mucho a completar después el diagrama de clases.

Como siempre, Ada tiene una solución.

-Tendremos que investigar los diagramas de secuencia.



En los **diagramas de secuencia**, los objetos/actores que forman parte del escenario de un caso de uso se representan mediante rectángulos distribuidos horizontalmente en la zona superior del diagrama, a los que se asocia una línea temporal vertical (una para cada actor) de las que salen, en orden, los diferentes mensajes que se pasan entre ellos.

Con esto el equipo de desarrollo puede hacerse una idea de las diferentes operaciones que deben ocurrir al ejecutarse una determinada tarea y el orden en que deben realizarse.

### Reflexiona

Los diagramas de secuencia completan a los diagramas de casos de uso, ya que permiten al equipo de desarrollo hacerse una idea de qué objetos participan en el caso de uso y como interactúan a lo largo del tiempo.

### 3.1.1.- Representación de objetos, línea de vida y paso de mensajes.

#### Caso práctico

Ada, orienta a su equipo:

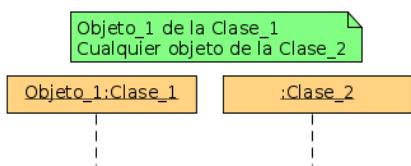
-Bien, ¿qué nos haría falta para poder representar la interacción de los objetos que participan en el caso de uso a lo largo del tiempo?

-Alguna manera de representar los objetos, el paso del tiempo y el paso de mensajes ¿no?



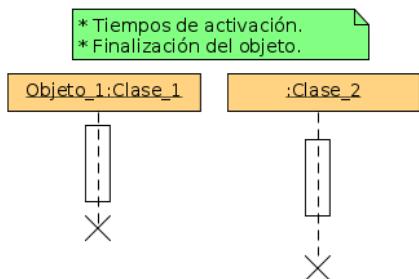
#### Representación de objetos y linea de vida.

En un **diagrama de secuencia**, los **objetos** se dibujan mediante rectángulos y se distribuyen horizontalmente en la parte superior del diagrama. Por cada objeto se identifica su nombre, seguido del símbolo de dos puntos y a continuación el nombre de la clase a la que pertenece. Si no se indica el nombre del objeto, se considera que para el propósito del diagrama es válido cualquier objeto de la clase.



De cada rectángulo sale una línea punteada que representa el paso del tiempo, se denomina **Línea de vida**. La línea de vida se prolonga mientras el objeto es relevante en el diagrama, una vez deja de serlo se indica mediante una cruz "X", dejando por tanto de existir a partir de ese momento.

Cuando el objeto toma protagonismo en el intercambio de mensajes, se dice que está **activo** y se indica mediante un recuadro sobre su línea de vida.

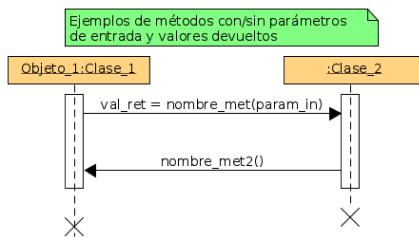


Una línea de vida puede estar encabezada por otro tipo de instancias como el sistema o un actor que aparecerán con su propio nombre. Usaremos el sistema para representar solicitudes al mismo, como por ejemplo pulsar un botón para abrir una ventana o una llamada a una subrutina

#### Paso de mensajes (Invocación de métodos).

Los **mensajes**, que significan la invocación de métodos, se representan como flechas horizontales que van de una línea de vida a otra, indicando con la flecha la dirección del mensaje. Los mensajes se dibujan desde el objeto que envía el mensaje al que lo recibe, pudiendo ser el mismo objeto emisor y receptor de un mensaje. El orden en el tiempo viene determinado por su posición vertical, un mensaje que se dibuja debajo

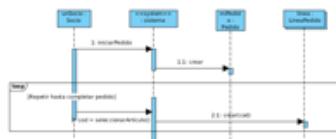
de otro indica que se envía después, por lo que no se hace necesario un número de secuencia. Los **mensajes** tendrán un **nombre** y pueden incluir **argumentos de entrada**, **valores devueltos** e **información de control** (condición o iteración).



Una notación alternativa para recoger valores devueltos por los métodos es dibujar una línea de puntos finalizada en flecha, que irá desde el objeto destinatario del mensaje al que lo ha generado, acompañado del texto del valor devuelto.

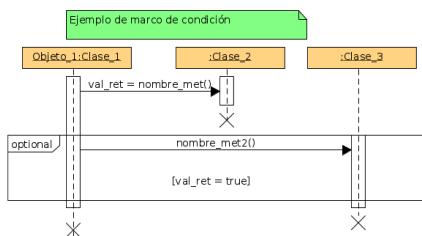
### Condicionales e iteraciones.

Además de presentar acciones sencillas que se ejecutan de manera secuencial también se pueden representar algunas situaciones más complejas como bucles usando marcos, normalmente se nombra el marco con el tipo de bucle a ejecutar y la condición de parada. También se pueden representar flujos de mensajes condicionales en función de un valor determinado.



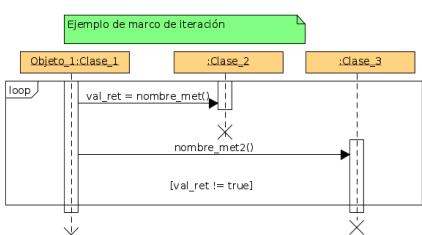
Las secuencias de control; tanto condicionales, como iterativas, se pueden representar usando **marcos**, normalmente se nombra el marco con el tipo de bucle a ejecutar y la condición de parada. También se pueden representar flujos de mensajes condicionales en función de un valor determinado.

La expresión a evaluar para la condición o iteración se representa entre corchetes.



Combinando varios marcos opcionales es posible representar diferentes alternativas en la ejecución de un diagrama de secuencia.

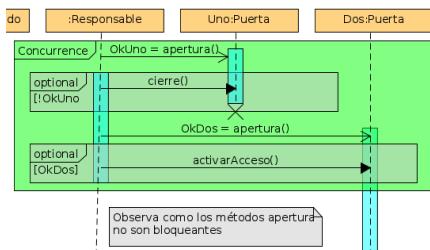
Para el caso de una iteración, tenemos el siguiente ejemplo.



Por defecto los métodos son **bloqueantes**, se entiende que el proceso del diagrama de secuencia completa cada método antes de continuar con el siguiente, es una secuencia de métodos en el tiempo. Pero en ocasiones se producen situaciones en las que se desea mostrar varios procesos en paralelo (**conurrencia**), se puede reflejar mediante el uso de marcos con la etiqueta **concurrence**.

Junto a los marcos de concurrencia, se hace necesario el uso de **métodos no bloqueantes (asíncronos)**, que permitan en paralelo activar diferentes procesos. La notación utilizada para los métodos asíncronos es

una línea finalizada con media cabeza de flecha o en UMLet una línea cuya punta flecha no está rellena.



Por último destacar que se puede completar el diagrama añadiendo **etiquetas** y **notas** en el margen izquierdo que aclare la operación que se está realizando.

## Autoevaluación

¿Cuál de estos elementos no forma parte de un diagrama de secuencia?

- Actor.
- Objeto.
- Bucle.
- Evento.

Incorrecto, pueden intervenir porque desencadenan acciones que producen la invocación de mensajes.

No es correcto, son una parte esencial de un diagrama de secuencia.

Falso, son útiles para representar acciones iterativas.

Así es, los eventos no tienen representación en el diagrama de secuencia que se centra más en el envío de mensajes.

## Solución

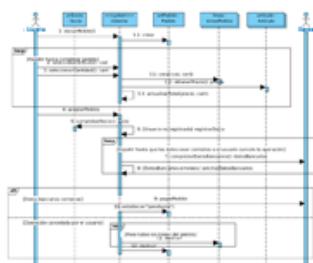
1. Incorrecto
2. Incorrecto
3. Incorrecto
4. Opción correcta

## 3.1.2.- Ejercicio resuelto 1 ("Generar pedido") (Elaboración de un diagrama de secuencias).

### Diagrama de interacción: Diagrama de secuencia

Vamos a generar el diagrama de secuencia que representa el caso de uso "Generar pedido" del diagrama de casos de uso del ejercicio resuelto 1 "ZAPATERÍA TACÓN DE ORO" (en el punto 2 de "Los diagramas de casos de uso"). En dicho diagrama se establece la secuencia de operaciones que se llevarán a cabo entre los diferentes objetos que intervienen en el caso de uso.

Este es el diagrama ya terminado, en el se han incluido todas las entidades (actores, objetos y sistema) que participan en el diagrama, y se han descrito todas las operaciones, incluidos los casos especiales, como es el registro de usuarios o la gestión de los datos bancarios. También incluye el modelado de acciones en bucle, como es la selección de artículos y de acciones regidas por condición, como es la posibilidad de cancelar el pedido si hay problemas con la tarjeta de crédito.



[Resumen textual alternativo](#)

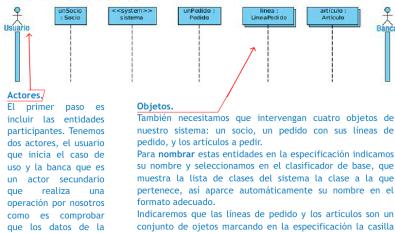
### Debes conocer

En la siguiente presentación puedes encontrar una descripción de como elaborar este diagrama con Visual Paradigm.

#### Crear el diagrama de secuencia

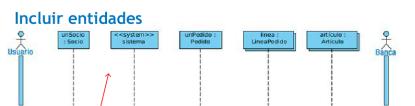
- El diagrama de secuencia que vamos a tratar es el del caso de uso Hacer pedido, que hemos descrito en el apartado anterior. Si no recuerdas bien cual era la descripción del caso te invito a que la repases en el punto 2.4 de los contenidos de la unidad.
- Este es un diagrama muy completo que incluye bastantes elementos de este diagrama, como bucles o condicionantes, veamos como incluirlos con la herramienta Visual Paradigm, como siempre, debes saber que puedes investigar y utilizar otras herramientas que sirvan para este mismo propósito.

#### Incluir entidades



#### Mensajes

Los mensajes pueden devolver un valor, que podemos escribir en la especificación del mismo en la opción return value.

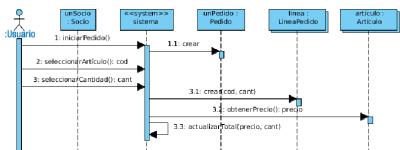


Sistema.  
Por último también interviene el sistema en si mismo, que utilizaremos para las operaciones relacionadas con la Interfaz gráfica y las que se lanzan directamente para crear objetos, recuperar información del sistema como los datos de un artículo, o comunicarse con entidades externas como la banca.

Se añade como una línea de vida más, a la que en su especificación indicamos que su nombre es sistema, y en la pestaña Esteriotipos seleccionamos System.

## Mensajes

En esta imagen vemos los mensajes correspondientes al apartado de añadir los datos del pedido. Se selecciona el artículo y la cantidad, se crea una nueva línea de pedido y se recuperan los datos del artículo para obtener suprecio y actualizar el total. Puesto que este proceso lo podemos repetir en más de una ocasión lo meteremos en un bucle.



## Mensajes

Para añadir un bucle seleccionamos la opción Loop Combined Fragment.

## Mensajes

Añadiremos condiciones de guarda cuando queramos indicar que un mensaje se enviará sólo si se cumple cierta condición, por ejemplo, sólo registraremos a un usuario si no es socio ya.



## Mensajes

También se pueden incluir condiciones más elaboradas que implique una bifurcación en el flujo de eventos, como es el caso de la comprobación de la tarjeta, si todo marcha bien se finalizará la creación del pedido, si no, el usuario puede cancelar la operación y terminar sin guardar nada.

### 3.1.3.- Ejercicio resuelto 2 ("ESTADIO") (Elaboración de un diagrama de secuencia).

#### Ejercicio Propuesto

Se pretende desarrollar un programa que dé respuesta a las necesidades de un estadio de fútbol en uno de sus partidos. Tras varias entrevistas con el responsable, se ha llegado al acuerdo de que los requisitos funcionales se pueden recoger en el caso de uso Gestión del estadio donde hay que considerar actividades como:

- **Control de puntos de acceso.** Se dispone de 2 puertas, a las que el responsable del estadio dará orden de apertura. Cada puerta hace un test interno y devuelve al responsable Ok o Ko. Si una puerta no abre, queda inactiva para el resto del partido. El proceso de apertura se realiza de forma simultánea en ambas puertas. Al finalizar el partido, el responsable del estadio dará orden de cierre de las puertas que se han usado durante el partido.
- **Control de acceso de aficionados.** Si la puerta está disponible, el responsable da orden de iniciar el acceso al estadio. Mientras haya aficionados, el operario de la puerta valida la entrada, si es correcta le da acceso al campo, en caso contrario avisa al responsable del estadio que ha habido un intento de acceso con entrada falsa.

**Nota:**

- Considera en este diagrama que la puerta uno está averiada y la puerta dos operativa.

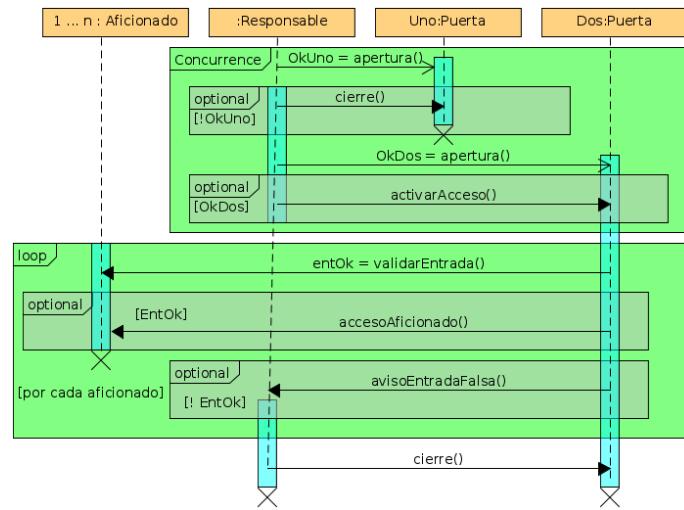
**Se pide:**

Desarrollar el diagrama de secuencia resultante del caso de uso planteado.

**Atiende a aspectos como:**

- Identificación de los objetos involucrados en el diagrama de secuencia.
- Que el diagrama recoja la secuencia de mensajes intercambiados, tomando en consideración las funcionalidades descritas en el enunciado.
- Uso correcto de la notación vista para este tipo de diagramas.
- Que los diagramas generados sean visualmente útiles. Un diagrama de secuencia debe dar una idea clara/rápida de la secuencia de acciones que se derivan de la ejecución del caso de uso que representa.

[Mostrar retroalimentación](#)



### 3.1.4.- Ejercicio resuelto 3 ("ROPERO") (Elaboración de un diagrama de secuencia).

## Ejercicio Propuesto

Se pretende crear un programa para la gestión de ciertas funciones de una discoteca. En particular, el depósito de los abrigos en el ropero.

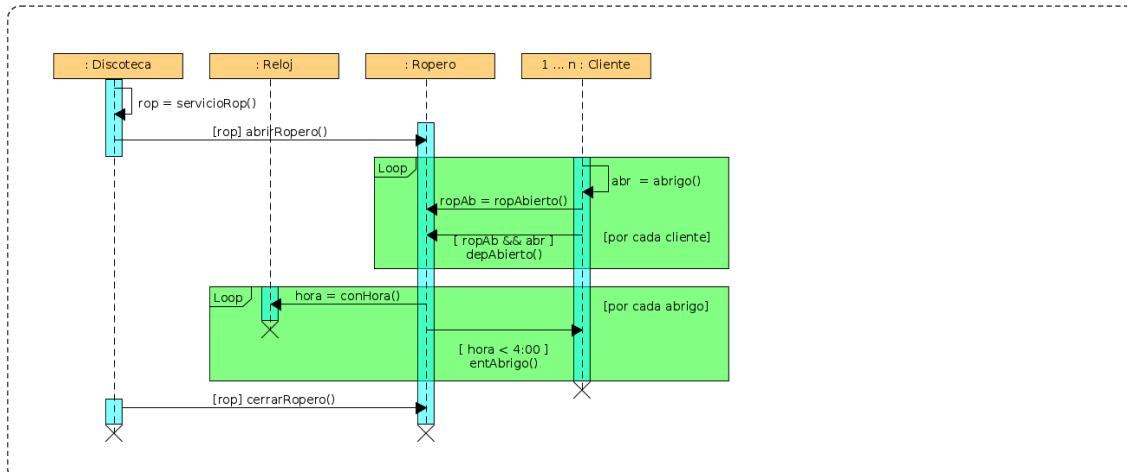
Algunas consideraciones:

- El responsable de la discoteca cada día decide si abre el vestíbulo, y es el encargado de su apertura y cierre.
- El propio ropero informa a los clientes si proporciona o no servicio de guardarropa.
- Si está abierto, a primera hora recoge los abrigos de los clientes que tengan abrigo.
- Al finalizar la jornada, mientras haya abrigos en el ropero se devuelven siempre que sea antes de las 4 de la mañana.
- Se pide desarrollar el diagrama de secuencia resultante del caso de uso planteado.

Considera aspectos como:

- Identificación de los objetos involucrados.
- Que el diagrama recoja la secuencia de mensajes intercambiados, tomando en consideración las funcionalidades descritas en el enunciado.
- Uso correcto de la notación vista para este tipo de diagramas.

[Mostrar retroalimentación](#)



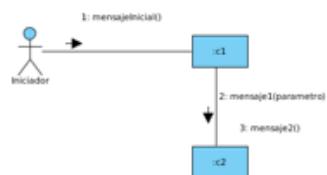
## 3.2.- Diagramas de colaboración.

### Caso práctico

-El diagrama de secuencia ha aportado información muy valiosa sobre la circulación de mensajes en los casos de uso, sin embargo estaría bien poder mostrar esta información de otra forma en la que se apreciase mejor el anidamiento de los mensajes, y el flujo de control entre objetos, ¿no creéis?



Al igual que los diagramas de secuencia, **los diagramas de colaboración** muestran una secuencia de ejecución de uno o varios casos de uso. La notación utilizada es muy similar y la principal diferencia radica en el modo de mostrar el orden de mensajes intercambiados entre objetos. Mientras el diagrama de secuencia establece el orden de los mensajes en el tiempo según su posición de arriba-abajo, el diagrama de colaboración lo hace mediante el etiquetado de los mensajes. Las interacciones entre los objetos se describen en forma de grafo en el que los nodos son objetos y las aristas son enlaces entre objetos a través de los cuales se pueden enviar mensajes entre ellos.



Los **diagramas de colaboración permiten una mejor organización visual de los objetos** al no ser obligada su representación en la parte superior del diagrama, en cambio la secuencia temporal suele ser más complicada de seguir.

**Los diagramas de colaboración tienen forma de grafo en el que los nodos son objetos y las aristas son los mensajes que intercambian.**

UMLet no dispone de herramientas para la elaboración de diagramas de colaboración directamente. No obstante, no resulta complicado generarlos a partir de los símbolos disponibles para otros diagramas: representación de objetos mediante cajas, paso de mensajes mediante líneas, información de los métodos mediante descripciones textuales; todos ellos disponibles en los diagramas de secuencia de UMLet.

### Reflexiona

Los diagramas de colaboración y secuencia utilizan los mismo elementos pero distribuyéndolos de forma diferente, ¿crees que son semejantes?

[Mostrar retroalimentación](#)

Es cierto, ambos diagramas representan la misma información, representan entidades del sistema y los mensajes que circulan entre ellas, además en ambos casos es fácil detectar la secuencialidad bien por el uso de líneas de vida, bien por los números de secuencia. De hecho en algunas aplicaciones para el desarrollo de estos diagramas es posible crear un diagrama a partir del otro.

## 3.2.1.- Representación de objetos.

### Caso práctico

-De acuerdo, mientras investigamos los diagramas de colaboración vamos a ver con un poco más de detalle qué significa la notación que se asigna a los objetos, ¿que diferencia hay entre usar los dos puntos o no hacerlo? ¿Podemos usar el nombre de una clase, solamente, o es obligatorio indicar el nombre del objeto?



Un objeto puede ser cualquier instancia de las clases que hay definidas en el sistema, aunque también pueden incluirse objetos como la interfaz del sistema, o el propio sistema, si esto nos ayuda a modelar las operaciones que se van a llevar a cabo.

Los objetos se representan mediante rectángulos en los que aparece uno de estos nombres.

- ✓ **NombreClase:** directamente se puede utilizar el nombre de la clase a la que pertenece el objeto que participa en la interacción. Pero esta representación hace referencia a la clase, el resto son objetos.
- ✓ **NombreObjeto:** se puede usar el nombre concreto del objeto que participa en la interacción, normalmente aparece subrayado.
- ✓ **:nombreClase:** cuando se coloca el símbolo ":" delante del nombre de la clase quiere decir que hace referencia a un objeto genérico de esa clase.
- ✓ **NombreObjeto:nombreClase:** hace referencia al objeto concreto que se nombre añadiendo la clase a la que pertenece.

Clase

:objeto

:Clase

objeto:clase

## 3.2.2.- Paso de mensajes.

### Caso práctico

-Y cuando enviamos un mensaje ¿cómo se representa exactamente?, ¿podemos incluir de alguna forma parámetros en los mensajes o valores devueltos? ¿Y si necesitamos indicar que el mensaje se enviará sólo si se cumple una determinada condición? ¿o que se envía dentro de un bucle?



Para que sea posible el paso de mensajes es necesario que exista una asociación entre los objetos, que se podrá mostrar mediante una línea que los une y una flecha que indique la dirección.

Al igual que sucedía en los diagramas de secuencia, es posible incluir parámetros en los mensajes, valores devueltos, mensajes enviados sólo si se cumple una determinada condición, o mensajes que se ejecutan varias veces (iteraciones).

La **sintaxis de un mensaje** es la siguiente:

```
[Secuencia] [*] [Condición] {valorDevuelto} : mensaje (argumentos de entrada) O  
[Secuencia] [*] [Condición] mensaje (argumentos de entrada) : {valorDevuelto}
```

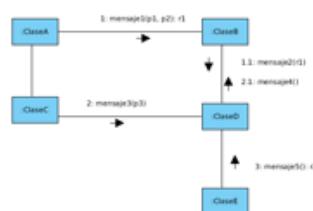
Donde:

- **Secuencia:** representa el nivel de anidamiento del envío del mensaje dentro de la interacción. Los mensajes se numeran para indicar el orden en el que se envían, y si es necesario se puede indicar anidamiento incluyendo subrangos.
- \*: indica que el mensaje es iterativo.
- **Condición de guarda:** debe cumplirse para que el mensaje pueda ser enviado.
- **ValorDevuelto:** lista de valores devueltos por el mensaje. Estos valores se pueden utilizar como parámetros de otros mensajes. Los corchetes indican que es opcional.
- **Mensaje:** nombre del mensaje.
- **Argumentos:** parámetros que se pasan al mensaje.

La enumeración de los mensajes se puede hacer de dos formas:

- Numeración simple: empieza en 1, se va incrementando en 1 y no hay ningún nivel de anidamiento.
- Numeración decimal: se muestran varios niveles de subíndices para indicar anidamiento de operaciones. Por ejemplo, 1 es el primer mensaje; 1.1 es el primer mensaje anidado en el mensaje 1, 1.2 es el segundo mensaje anidado en el mensaje 1; y así sucesivamente.

Como se ve en el ejemplo, se puede usar la misma asociación para enviar varios mensajes. Vemos que hay dos mensajes anidados, el 1.1 y el 2.1, se ha usado el nombre de los mensajes para indicar el orden real en el que se envían.



Los mensajes 1, 1.1 y 2 tienen parámetros y los mensajes 1 y 3 devuelven un resultado.

Se contempla la bifurcación en la secuencia añadiendo una condición en la sintaxis del mensaje:

[Secuencia][\*][CondiciónGuarda]{valorDevuelto} : mensaje (argumentos)

Cuando tenemos una condición se repite el número de secuencia y se añaden las condiciones necesarias, como vemos en la imagen según la condición se enviará el mensaje 1 o el 2, pero no ambos, por lo que coinciden en número de secuencia.

La iteración se representa mediante un \* al lado del número de secuencia, pudiendo indicarse entre corchetes la condición de parada del bucle.

**Nota:** VP-UML modifica el orden en el que aparecen los datos pero no su notación.

## Autoevaluación

Indica qué afirmación no es correcta para el siguiente diagrama:



- El objeto ob2 es multiobjeto.
- Se envía un mensaje del objeto 1 al objeto 2.
- El mensaje operacion(pp) se ejecutará siempre.
- La operación se puede ejecutar varias veces.

Esta afirmación es correcta, ya que el icono doble indica que así es.

Así es, según la dirección de la flecha.

Efectivamente, esta afirmación no es correcta, ya que la operación solo se ejecuta si se cumple la condición de guarda.

Esta afirmación es cierta, el asterisco que la precede así lo indica.

## Solución

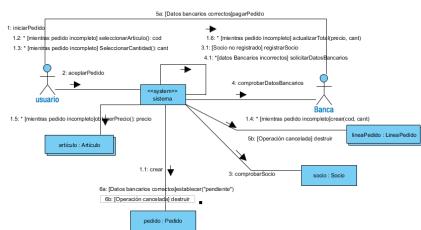
1. Incorrecto
2. Incorrecto
3. Opción correcta
4. Incorrecto

### 3.2.3.- Ejemplo de un diagrama de colaboración.

#### Diagrama de interacción: Diagrama de colaboración

A continuación se muestra un diagrama de colaboración de ejemplo.

Este es el diagrama de colaboración que representa el caso de uso "Generar pedido" del diagrama de casos de uso del ejercicio resuelto 1 "ZAPATERÍA TACÓN DE ORO" (en el punto 2 de "Los diagramas de casos de uso"). Se ha creado siguiendo el diagrama de secuencia, por lo que no te debe ser muy difícil seguirlo, de hecho algunas aplicaciones para la creación de estos diagramas permiten la obtención de uno a partir de otro. Debes tener en cuenta que la aplicación modifica un poco la signatura de los mensajes, el valor devuelto se representa al final precedido de dos puntos.



[Resumen textual alternativo](#)

Los aspectos más destacados son los siguientes:

- ✓ Las actividades que se repiten o pueden repetirse se marcan con un asterisco y su condición.
- ✓ Las condiciones de guarda se escriben en el mismo nombre del mensaje.
- ✓ El flujo alternativo de eventos según si el usuario cancela el pedido o no, obliga a modificar los números de secuencia de los mensajes 5 y 6, pasando a tener los mensajes 5a y 6a y 5b y 6b, según la condición. Puedes modificar el número de secuencia de los mensajes abriendo la especificación del diagrama, y seleccionando la pestaña Mensajes, donde puedes editar los números de secuencia haciendo doble clic sobre ellos.
- ✓ Al objeto "sistema" se le ha asignado el estereotipo system.

## 4.- Diagramas de estados.

### Caso práctico

Ada espera que su equipo continúe con tan buen ánimo para estudiar un tipo de diagrama más, que completará las diferentes visiones de la dinámica de un sistema que proporciona UML. Son los diagramas de estados, que les permitirán analizar cómo va cambiando el estado de los objetos que tienen una situación variable a lo largo del tiempo.



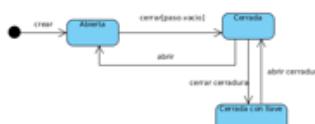
Los **diagramas de estados** permiten analizar como va evolucionando el **estado de un objeto** a lo largo del tiempo, es decir, representa su comportamiento transitando por una serie de estados.

Modelan el **comportamiento dinámico** de los objetos en respuesta a determinados **eventos**.

En relación con el diagrama de estados se cumple que:

- Un **objeto** está en un **estado concreto** en un cierto momento, que principalmente viene determinado, por los **valores de sus atributos**.
- La **transición de un estado** a otro es momentánea y se produce cuando ocurre un determinado **evento**.

Por ejemplo, aquí tenemos el **diagrama de estados de una puerta**.



### Autoevaluación

Analiza el diagrama de estados de la puerta, según está dibujado, ¿se puede abrir una puerta que está cerrada con llave directamente?

- Verdadero.
- Falso.

No ya que no hay una transición directa desde cerrada con llave a abierta, para poder abrirla necesitamos abrir la cerradura primero.

Cierto, antes hay que abrir la cerradura.

## Solución

1. Incorrecto
2. Opción correcta

## 4.1.- Estados y eventos.

### Caso práctico

Ada indica a su equipo que para entender bien la dinámica de un diagrama de estados deben comenzar por analizar sus componentes fundamentales: estados y eventos.



Un **estado** es una situación en la vida de un objeto en la que satisface cierta condición, realiza alguna actividad o espera algún **evento**.

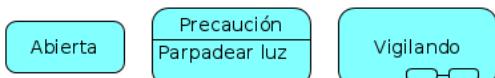
Existen **tres tipos de estado** en los que se puede encontrar un objeto:

- **Estado inicial.** Punto de partida por defecto del diagrama de estados. Corresponde a los valores de los atributos de una clase en el momento de instanciar un nuevo objeto.
- **Estado final.** Estado en el que se encuentra el objeto una vez finalizada la secuencia de eventos que pueden proporcionar transiciones entre estados.
- **Estado intermedio.** Cualquiera de los estados intermedios entre los dos anteriores.



Los estados se representan mediante una caja y admite algunas variantes. La información que se muestra en los estados suele ser:

- **Nombre del estado.** Por ejemplo Abierta.
- **Nombre del estado y acción/actividad asociada al objeto** mientras se encuentra en ese estado. En un semáforo en estado de precaución, se produce la actividad de parpadeo de la luz.
- **Estado con subestados.** En el ejemplo se indica que el estado vigilando tiene asociado una serie de subestados, si se trata de un vigilante de seguridad, el estado "vigilando" podría tener relacionados los subestados de ruta a pie y/o de visionado de cámaras.



Un **evento** es un acontecimiento que dispara una transición entre dos estados del objeto. Existen eventos externos y eventos internos según el agente que los produzca.

**Tipos de eventos:**

- **Señales (excepciones):** la recepción de una señal, producida por una **situación excepcional en el sistema**. Puede ser origen de una transición entre estados.
- **Llamadas:** la recepción de una petición para **invocar una operación**. Normalmente un evento de llamada es **manejado por un método del objeto**.
- **Paso de tiempo:** el evento se genera como consecuencia del cumplimiento de un temporizador.
- **Cambio de estado:** evento generado por un cambio en el estado o el cumplimiento de una condición.

## 4.2.- Transiciones.

### Caso práctico

-De acuerdo, los estados son situaciones específicas en las que se puede encontrar un objeto, y los eventos pueden hacer que un objeto cambie de estado, y, ¿cómo representamos eso?



Una **transición** de un estado A a un estado B, se produce cuando se origina el evento asociado y se satisface cierta condición especificada, en cuyo caso se ejecuta la acción de salida de A, la acción de entrada a B y la acción asociada a la transición.

La notación de una transición tiene tres partes, todas ellas optativas:

Evento(argumentos) [Condición] / Acción.

#### Elementos de una transición:

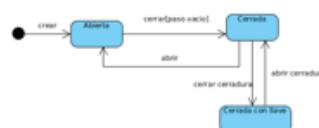
- **Evento:** cuando se produce un evento, afecta a todas las transiciones que lo contienen en su etiqueta.
- **Condición:** expresión evaluable como verdadera o falsa. Si es falsa, la transición no se dispara.
- **Acción:** conjunto de actuaciones que lleva asociada la transición. Puede incluir llamadas a operaciones de objetos, creación o destrucción de objetos ...

**Ejemplo:** Vamos a ver el **diagrama de estados para un semáforo**. Recoge ejemplos de los tres elementos descritos para las transiciones.



### Autoevaluación

Recordemos el diagrama de estado de la puerta:



¿Qué significa la **signatura de la transición "cerrar [paso.vacio]"?**

- Que cuando cerremos la puerta el paso quedará vacío.
- Que para cerrar la puerta el paso debe estar vacío.

- Que cuando se está cerrado la puerta se vacía el paso.

No es así, los corchetes no representan una acción a la salida.

Así es, los corchetes en un diagrama UML significan una condición que se debe cumplir. No se podrá cerrar la puerta hasta que el paso no esté vacío.

No es cierto, la acción a realizar se representa con un /.

## Solución

1. Incorrecto
2. Opción correcta
3. Incorrecto

## 4.3.- Ejercicio resuelto 1 ("Generar pedido") (Elaboración de un diagrama de estados).

### Diagrama de estados:

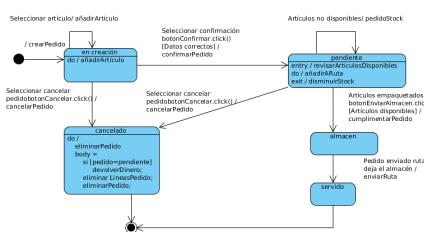
Para exemplificar la creación de un diagrama de estados vamos a ver el que representa el caso de uso "**Generar pedido**" del diagrama de casos de uso del ejercicio resuelto 1 "ZAPATERÍA TACÓN DE ORO" (en el punto 2 de "Los diagramas de casos de uso") que cumple con las condiciones que hemos visto al principio, tiene un comportamiento significativo en tiempo real, ya que su situación tanto física, como el sistema, va evolucionando conforme pasa el tiempo, y participa en varios casos de uso (como Hacer pedido y Cumplimentar pedido).

Los diferentes estados en los que puede estar un pedido son:

- ✓ **En creación:** es cuando se están seleccionando los productos que formará el pedido.
- ✓ **Pendiente:** está en este estado desde que se confirma el pedido hasta que se selecciona para preparar su envío.
- ✓ **En almacén:** está en este estado cuando es elaborado el paquete y se ha asignado a una ruta, hasta que se envía a través de la ruta que le corresponde.
- ✓ **Servido:** Cuando el pedido es enviado. En este caso se envía una señal física desde el almacén cuando el transporte abandona el almacén.
- ✓ **Cancelado:** puede llegarse a esta situación por dos motivos, o bien se cancela mientras se está haciendo por problemas con la tarjeta de crédito, o bien porque, una vez pendiente de su gestión el usuario decide cancelarlo, la diferencia fundamental entre ambos es que en el segundo caso hay que devolver el importe pagado por el pedido al socio que lo ha comprado.

Las transiciones entre estados se producen por llamadas a procedimientos en todos los casos, no intervienen cambios de estado o el tiempo, ni señales.

El diagrama quedaría de la siguiente manera:



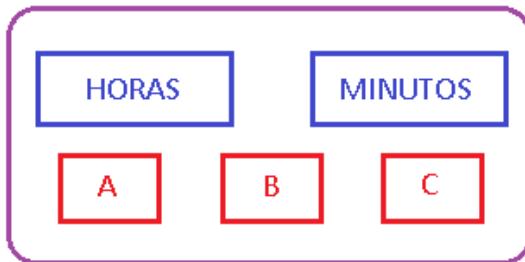
A los **estados** se les ha añadido la acción a realizar, apartado do/ y en algunos casos la acción de entrada, por ejemplo en el caso del estado pendiente, se debe revisar que los artículos a enviar tengan disponibilidad y la de salida, en el ejemplo disminuir el stock.

**Nota:** para incluir las condiciones de guarda en el diagrama debes rellenar el apartado "Guard" de la especificación, si necesitas añadir alguna acción puedes hacerlo rellenando el apartado "Effect". Los eventos de disparo.

## 4.4.- Ejercicio resuelto 2 ("RELOJ")(Elaboración de un diagrama de estados).

### Ejercicio Propuesto

La siguiente figura muestra un **reloj digital** cuyo comportamiento se describe a continuación:



El reloj se enciende y está visualizando las horas y minutos.

Funciones de reloj:

Pulsado de A durante tres segundos: parpadea la hora. Para evitar cambios de hora involuntarios, si el tiempo de pulsado es inferior a tres segundos no se activa la función.

El botón B no funciona, si no se ha pulsado antes el botón A durante 3 segundos.

De tal forma que si el reloj está en el estado en el que la hora está parpadeando:

1. Si se pulsa el botón B incrementa la hora en una unidad.
2. Si se pulsa el botón A, pasará al estado de poder cambiar los minutos. Los minutos parpadearán. No se precisa mantener pulsado el botón porque se entiende que se está modificando la hora de forma voluntaria.

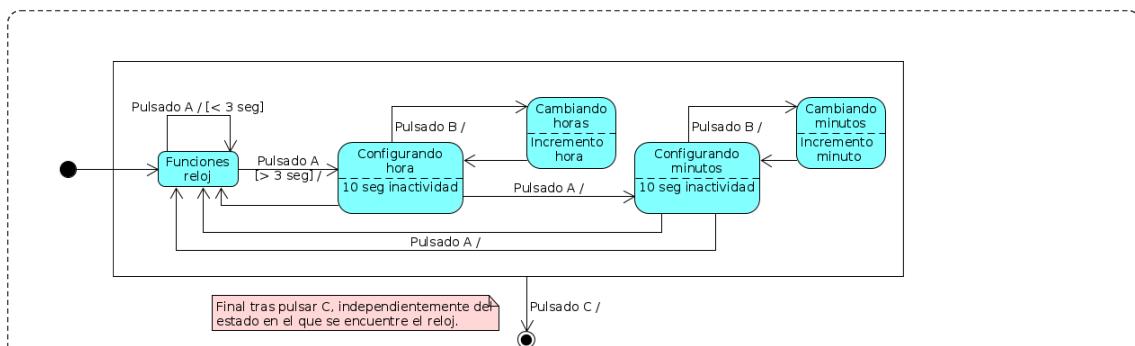
Si está el reloj en el estado de poder cambiar los minutos:

1. Pulsado del botón B: cada pulsación del botón B incrementa los minutos en una unidad
2. Si pulsamos el botón A, finaliza el modo configuración y vuelve a mostrar la hora.

Pulsado de C: apaga del reloj sin tener en consideración el estado en el que se encuentre.

Cuando el reloj está en modo configuración de horas o minutos, tras 10 segundos de inactividad abandona la configuración y pasa a modo funciones de reloj.

Mostrar retroalimentación



Notas:

- Las transiciones sin eventos, aunque pueden tener condiciones, se producen al finalizar la actividad del estado.
- Es posible salir de un estado tanto por un evento externo, como por el fin de la actividad del estado.
- El evento pulsado C genera el apagado del reloj con independencia del estado en el que se encuentre.

## 4.5.- Ejercicio resuelto 3 ("VIDA LABORAL") (Elaboración de un diagrama de estados).

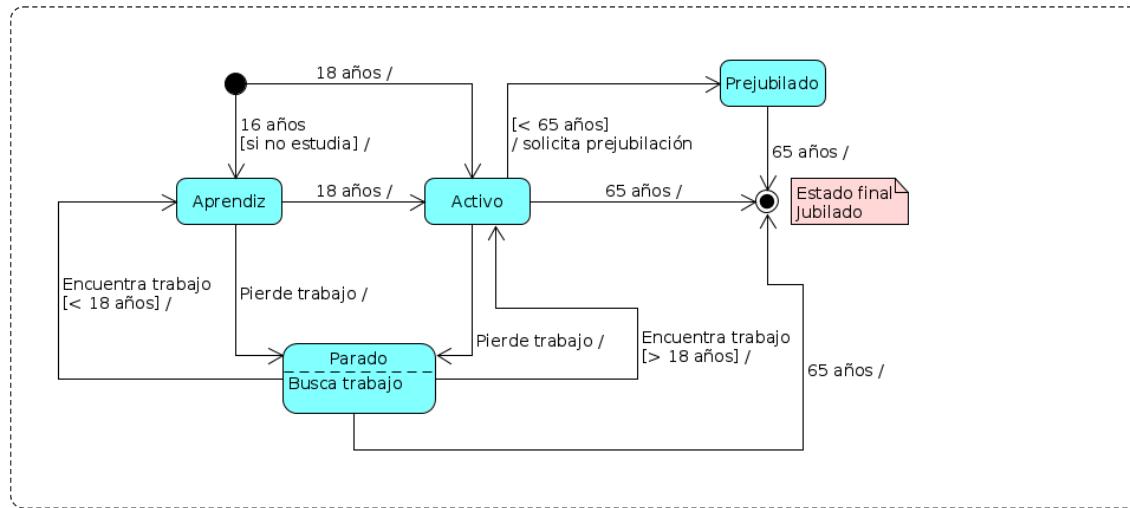
### Ejercicio Propuesto

Crea un diagrama de estados que muestre la **evolución de un empleado** a lo largo de su vida laboral.

Se considera que el proceso transcurre entre los 16 y 65 años de edad y se plantean los siguientes **estados**:

- **Preempleado.** Anterior a los 16 años. Se considera el estado inicial.
- **Aprendiz.** Es el periodo comprendido entre los 16 y 18 años para aquellas personas que han decidido no continuar sus estudios.
- **Activo.** El trabajador se encuentra en activo y con contrato en vigor.
- **Parado.** El trabajador ha perdido el empleo, su tarea principal es la búsqueda de un nuevo trabajo.
- **Prejubilado.** El trabajador solicita dejar de estar activo, pero no ha alcanzado la edad de 65 años. Desde el estado de parado no se considera la opción de solicitar la prejubilación.
- **Jubilado.** El trabajador ha cumplido los 65 años y pasa a disfrutar de un merecido descanso. Se considera el estado final.

[Mostrar retroalimentación](#)



## 5.- Diagramas de actividad.

### Caso práctico

Por el momento el equipo de BK no ha tenido problema en seguir lo que Ada les cuenta sobre los diagramas UML. Antonio, que está verdaderamente interesado en el tema hace a Ada la siguiente pregunta:

-¿Que pasaría si quisiera representar sólo las acciones que tienen lugar, prescindiendo de quien las genera, solo el flujo de la actividad del sistema, qué pasa primero, qué ocurre después y qué cosas pueden hacerse al mismo tiempo?

-Pasaría que tendrías que hacer un diagrama de actividad.



El **diagrama de actividad** es una especialización del diagrama de estados, organizado en torno a las **acciones** en lugar de los objetos, que se compone de una serie de actividades y representa como se pasa de unas a otras. **Las actividades se enlazan por transiciones automáticas**, es decir, cuando una actividad termina se desencadena el paso a la siguiente.

El diagrama de actividades resulta útil cuando se quiere representar sólo las acciones que tienen lugar, prescindiendo de quien las genera. ¿Qué pasa primero, qué ocurre después y qué cosas pueden hacerse al mismo tiempo?.

Se utilizan fundamentalmente para modelar el flujo de control entre actividades en el que se puede distinguir cuales ocurren **secuencialmente** a lo largo del tiempo y cuales se pueden llevar a cabo **concurrentemente**. Permite visualizar la dinámica del sistema desde otro punto de vista que complementa al resto de diagramas.

Un diagrama de actividades es un grafo conexo en el que los nodos son **estados**, que pueden ser de **actividad** o de **acción** y los arcos son **transiciones** entre estados.

### Reflexiona

¿Por qué decimos que el diagrama de actividades visualiza el comportamiento desde otro punto de vista del resto de diagramas?

[Mostrar retroalimentación](#)

En los anteriores diagramas, tratábamos la interacción entre objetos y entidades, cual es el flujo de mensajes, en qué orden y bajo qué condiciones se envían, en los diagramas de actividades se incluye el factor de la concurrencia, y trata sobre todo de expresar el flujo de las actividades, su elemento fundamental, y como se pasa de unas a otras. Además también representa como se influye en los objetos.

## 5.1.- Elementos del diagrama de actividad.

### Caso práctico

-Vale estoy preparado, ¿qué necesito para tener un diagrama de actividad?

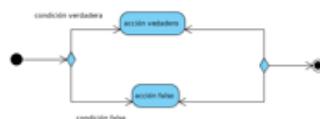


Normalmente los diagramas de actividades contienen:

- ✓ **Estados** de actividad y estados de acción.
  - ⇒ **Estado de actividad**: Elemento compuesto cuyo flujo de control se compone de otros estados de actividad y de acción.
  - ⇒ **Estado de acción**: Estado que representa la ejecución de una acción atómica, que no se puede descomponer ni interrumpir, normalmente la invocación de una operación. Generalmente se considera que su ejecución conlleva un tiempo insignificante.
  - ⇒ Pueden definirse también otro tipo de estados:
    - **Inicial**.
    - **Final**.



- ✓ **Transiciones**: Relación entre dos estados que indica que un objeto en el primer estado realizará ciertas acciones y pasará al segundo estado cuando ocurra un evento específico y satisfaga ciertas condiciones. Se representa mediante una línea dirigida del estado inicial al siguiente. Podemos encontrar diferentes tipos de transacciones:
  - ⇒ **Secuencial o sin disparadores**: Al completar la acción del estado origen se ejecuta la acción de salida y, sin ningún retraso, el control sigue por la transición y pasa al siguiente estado.
  - ⇒ **Bifurcación(Decision node)**: Especifica caminos alternativos, elegidos según el valor de alguna expresión booleana. Las condiciones de salida no deben solaparse y deben cubrir todas las posibilidades (puede utilizarse la palabra clave `else`). Pueden utilizarse para lograr el efecto de las iteraciones.

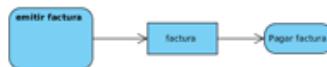


- ⇒ **Fusión (Merge node)**: Redirigen varios flujos de entrada en un único flujo de salida. No tiene tiempo de espera ni sincronización.
- ⇒ **División (Fork node)**: Permiten expresar la sincronización o ejecución paralela de actividades. Las actividades invocadas después de una división son concurrentes.



- ⇒ **Unión (Join node)**: Por definición, en la unión los flujos entrantes se sincronizan, es decir, cada uno espera hasta que todos los flujos de entrada han alcanzado la unión.
- ✓ **Objetos**: Manifestación concreta de una abstracción o instancia de una clase. Cuando interviene un objeto no se utilizan los flujos de eventos habituales sino flujos de objetos (se representan con una flecha de igual manera) que permiten mostrar los objetos que participan dentro del flujo de control

asociado a un diagrama de actividades. Junto a ello se puede indicar cómo cambian los valores de sus atributos, su estado o sus roles.



Se utilizan carriles o calles para ver **QUIENES** son los responsables de realizar las distintas actividades, es decir, especifican qué parte de la organización es responsable de una actividad.

- ✓ Cada calle tiene un nombre único dentro del diagrama.
- ✓ Puede ser implementada por una o varias clases.
- ✓ Las actividades de cada calle se consideran independientes y se ejecutan concurrentemente a las de otras calles.

## Autoevaluación

**Los diagramas de actividades, a diferencia del resto, permiten incluir la concurrencia en la representación del diagrama.**

- Verdadero.
- Falso.

Pues así es. Podemos representar acciones concurrentes utilizando las herramientas de fusión y unión, ya que se considera que todas aquellas acciones que queden entre ambas son concurrentes.

No es cierto, podemos representar concurrencia utilizando las herramienta de fusión y unión.

## Solución

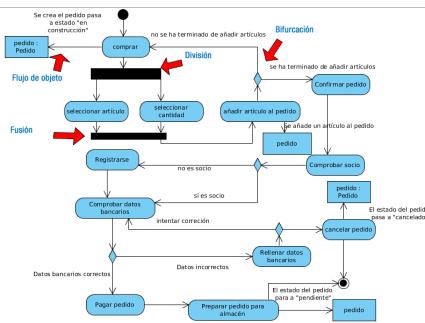
1. Opción correcta
2. Incorrecto

## 5.2.- Ejemplo de un diagrama de actividad.

### Diagrama de actividad:

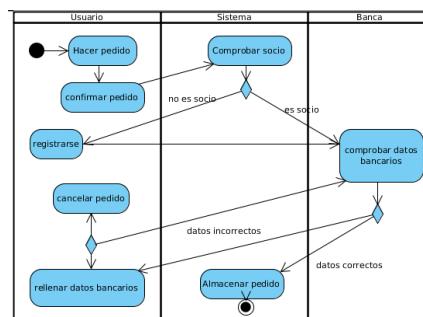
El siguiente diagrama de actividad representa el caso de uso "Generar pedido" del diagrama de casos de uso del ejercicio resuelto 1 "ZAPATERÍA TACÓN DE ORO" (en el punto 2 de "Los diagramas de casos de uso"), en el aparecen los elementos que hemos visto en las secciones anteriores.

- ✓ En las bifurcaciones se ha añadido la condición que indica si se pasa a una acción o a otra.
- ✓ Las acciones Seleccionar artículo y Seleccionar cantidad se han considerado concurrentes.



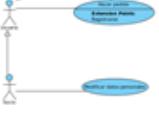
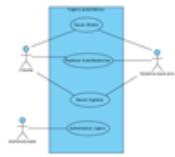
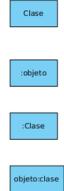
#### Resumen textual alternativo

En este otro diagrama se simplifican las acciones a realizar y se eliminan los objetos para facilitar la inclusión de calles que indican quien realiza cada acción:



**Nota:** Para añadir las calles en Visual Paradigm se utiliza la herramienta del panel "Vertical Swimlane".

# Anexo I.- Licencias de recursos.

Recurso (1)	Datos del recurso (1)	Recurso (2)	Datos del recurso (2)
	Autoría: Ministerio de Educación. Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.		Autoría: Ministerio de Educación. Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.
	Autoría: Ministerio de Educación. Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.		Autoría: Ministerio de Educación. Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.
	Autoría: Ministerio de Educación. Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.	 <b>Actor</b>	Autoría: Ministerio de Educación. Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.
	Autoría: María José Navascués González. Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.		Autoría: María José Navascués González. Licencia: Uso Educativo no comercial. Procedencia: Elaboración Propia.
	Autoría: María José Navascués González. Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.		Autoría: María José Navascués González. Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.
	Autoría: María José Navascués González. Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.		Autoría: María José Navascués González. Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.
	Autoría: María José Navascués González. Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.		Autoría: María José Navascués González. Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.
	Autoría: María José Navascués González. Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.		Autoría: María José Navascués González. Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.





<ul style="list-style-type: none"> <li>●</li> <li>●</li> <li><span style="border: 1px solid #ccc; padding: 2px;">Abierta</span></li> </ul>	<p>Autoría: Ministerio de Educación. Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>		<p>Autoría: Ministerio de Educación. Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>
	<p>Autoría: Ministerio de Educación. Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>		<p>Autoría: Ministerio de Educación. Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>
	<p>Autoría: Ministerio de Educación. Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>	<p><b>Primeros pasos</b> Este es un diagrama que muestra los primeros pasos que tienen que hacer los administradores para crear una estrategia de recaudación. Los pasos son: 1. Identificar las necesidades de la institución y establecer objetivos claros. 2. Recopilar información relevante sobre la situación actual de la institución. 3. Analizar la información recopilada y identificar las fortalezas y debilidades. 4. Proporcionar recomendaciones para mejorar la situación.</p>	<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>
	<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>	<p><b>Identificar Funcionalidades</b> Para facilitar la creación del diagrama, se ha separado la funcionalidad en tres categorías principales: 1. Funcionalidades del usuario: Permite al usuario crear y editar la primera que publica sobre el análisis que hace en el sistema. 2. Funcionalidades del administrador: Permite al administrador crear y editar la primera que publica sobre el análisis que hace en el sistema. 3. Funcionalidades del responsable de alcance: Permite al responsable de alcance crear y editar la primera que publica sobre el análisis que hace en el sistema.</p>	<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>
	<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>		<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>
	<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>		<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>
	<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>		<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>
	<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>		<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>
	<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>		<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>
	<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>	<p><b>Funcionalidades del responsable de alcance</b> El responsable de alcance tiene la función de gestionar los contenidos de la web, así como la gestión de las estrategias de recaudación. Puede crear y editar las estrategias existentes y dar de alta nuevas estrategias. También tiene la función de revisar las estrategias que ya están en el sistema y de eliminar las estrategias que ya no son necesarias.</p>	<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>
	<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>	<p><b>Crear el diagrama de secuencia</b> Este es un diagrama que muestra el proceso de creación de un diagrama de secuencia. Los pasos son: 1. Seleccionar el tipo de diagrama de secuencia que se desea crear. 2. Definir los nodos y las transiciones entre ellos. 3. Añadir acciones y condicionales a los nodos. 4. Validar el diagrama para asegurarse de que es correcto.</p>	<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>
	<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>		<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>

<p><b>Misiones</b> Las misiones tienen como objetivo principal contribuir al desarrollo integral de la persona, en el marco de la formación integral del ciudadano chileno.</p>	<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>		<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>
<p><b>Misiones</b> Las misiones tienen como objetivo principal contribuir al desarrollo integral de la persona, en el marco de la formación integral del ciudadano chileno.</p>	<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>		<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>
<p><b>Misiones</b> Las misiones tienen como objetivo principal contribuir al desarrollo integral de la persona, en el marco de la formación integral del ciudadano chileno.</p>	<p>Autoría: Ministerio de Educación Licencia: Uso educativo no comercial. Procedencia: Elaboración propia.</p>		