

# UT04.-DISEÑO ORIENTADO A OBJETOS. DIAGRAMAS DE COMPORTAMIENTO.

# Índice

## 1.- Diagramas de comportamiento

## 2.- Diagramas de casos de uso

### 2.1 Qué son

### 2.2 Elementos

### 2.3 Actores

### 2.4 Casos de uso

### 2.5 Relaciones

### 2.6 Estereotipos

### 2.7 Elaboración

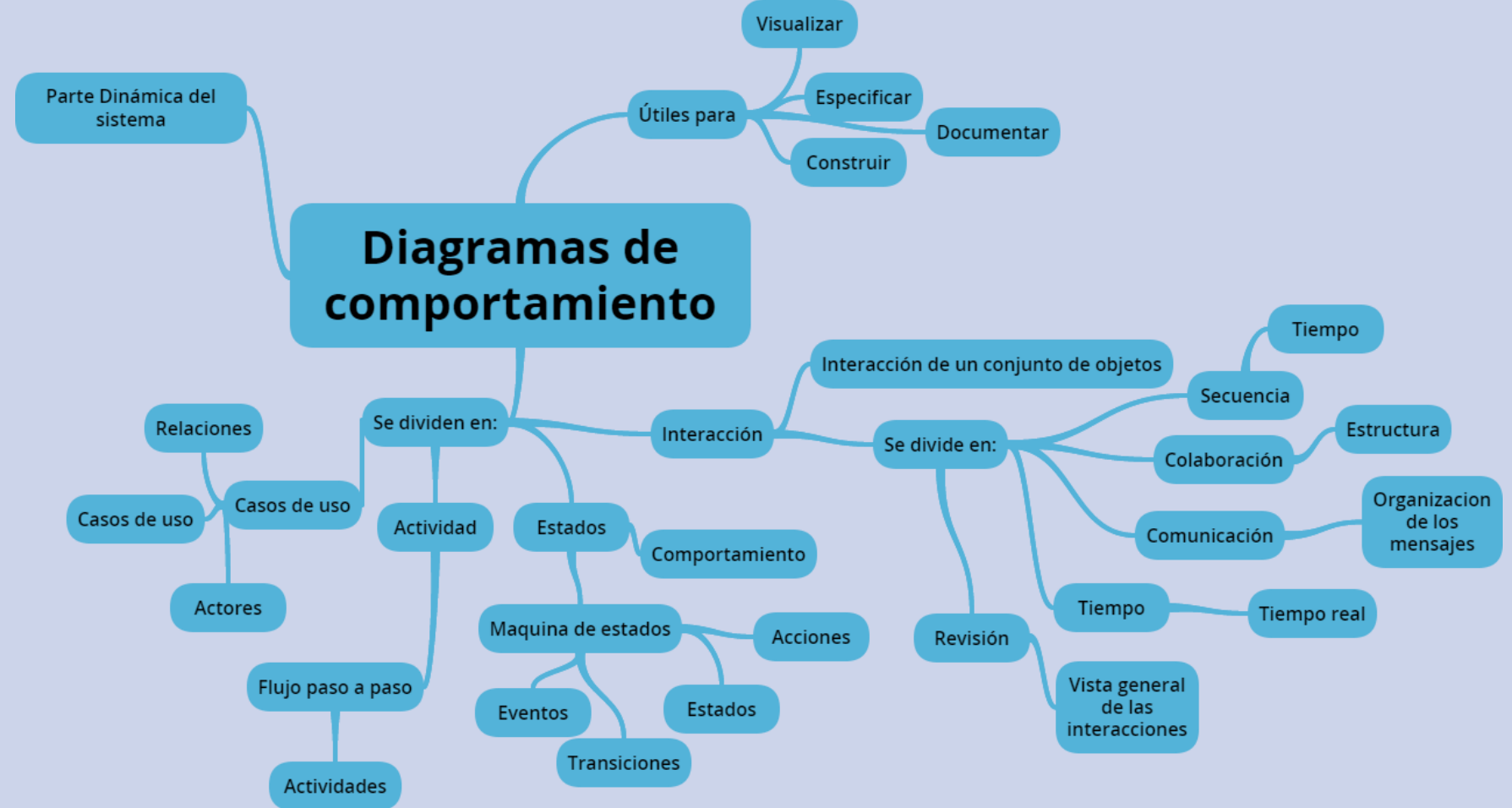
### 2.8 Escenarios

### 2.9 Buenas prácticas

## 3.- Diagramas de interacción

### 3.1.- Diagramas de secuencia.

### 3.2.- Diagramas de colaboración.



## 1. Diagramas de comportamiento.

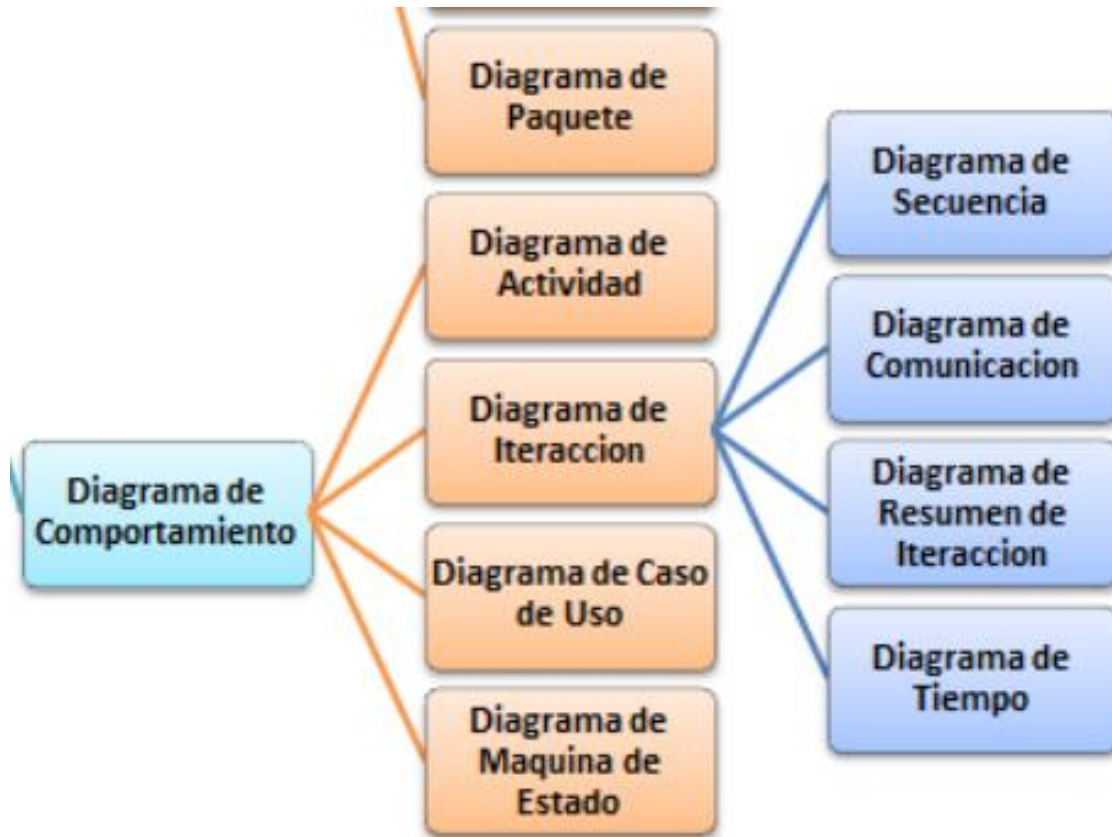
# Introducción.

Etapas del  
desarrollo de  
una aplicación

- Análisis → Especificación de requisitos del software
- Diseño
  - Diagramas de clases
  - Algoritmos
- Codificación
- Pruebas
- Documentación
- Explotación y mantenimiento

- Casos de uso
- DFD

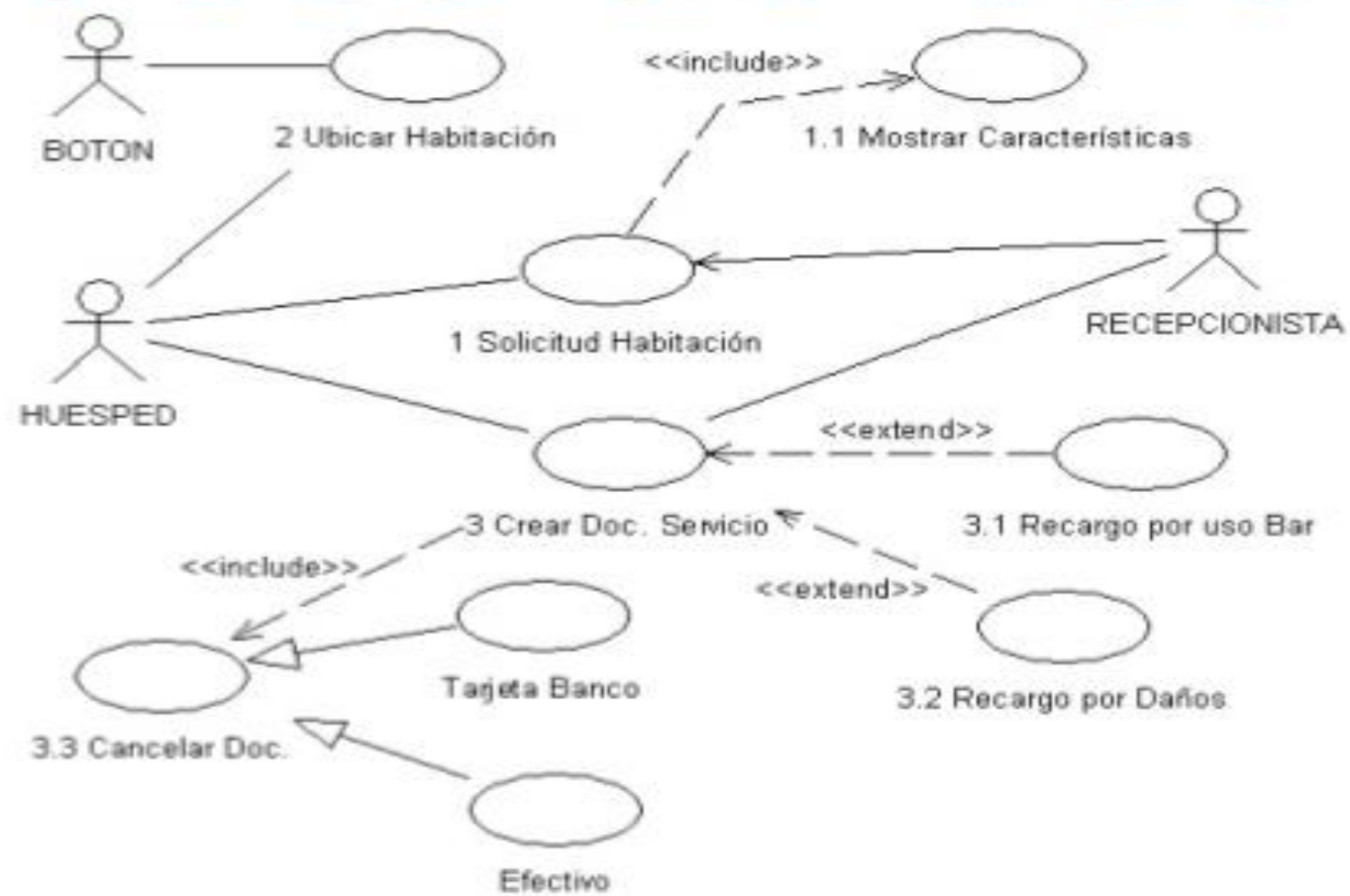
# Introducción.



- **Diagramas 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.
- **Diagramas de comportamiento:** nos dan una visión dinámica del sistema, son una representación del modelo funcional del sistema.

# Introducción.

- **Diagramas de casos de uso:** describen el **comportamiento** del sistema desde el punto de vista de los usuarios que interactúan con él.
- **Diagramas de actividad:** **modelan el flujo de un caso de uso**. Similares a los diagramas de flujo, muestran los pasos, bifurcaciones, decisiones ...
- **Diagramas de estado:** Estados por los que pasa un objeto.
- **Diagramas de interacción,** también utilizados en el **análisis de requisitos:**
  - **Diagramas de colaboración:** prima la relación estructural de los objetos que participan en la interacción.
  - **Diagramas de secuencia:** prima la secuencia temporal de los mensajes intercambiados entre los objetos que conforman la interacción.
  - **Diagramas de tiempos:** definen el comportamiento de diferentes objetos dentro de una escala de tiempo.
  - **Diagramas de vista de interacción:** muestra la cooperación entre otros diagramas de interacción.



## 2. Diagramas de casos de uso.

# Diagramas de casos de uso

- **Definición formal:** Un **caso de uso** especifica un conjunto de secuencias de acciones, incluyendo variantes, que el sistema puede ejecutar y que produce un resultado observable, de valor para un particular actor.
- Define **qué** hace el sistema, **no cómo** lo hace.



# Diagramas de casos de uso

- **Definen los requisitos funcionales del sistema** (funciones del sistema) *durante la etapa de análisis*, diseñando un **conjunto de escenarios** que faciliten una descripción de cómo se usará el sistema.
  - **Determina qué** puede hacer cada tipo de usuario con el sistema, expresado de **forma comprensible para el cliente**.
  - **Facilitan** la comunicación entre el equipo de desarrollo y el cliente que demanda la solución software → **ventaja** → facilidad de interpretarlos.
- Se usan también en la **fase de pruebas**, para verificar que el sistema cumple los requisitos funcionales.

# Diagramas de casos de uso

- **Modelan el comportamiento del sistema** desde el punto de vista del usuario que interactúa con el sistema.
- Técnica para definir los **requisitos funcionales** del software:
  - Representación gráfica de los requisitos funcionales.
  - Descripción de los requisitos funcionales.

# Diagramas de casos de uso

---

- A partir de los diagramas de casos de uso se desprenden otros (normalmente se realiza antes del diagrama de clases), que describen tanto la estructura del sistema como su comportamiento.

# Diagramas de casos de uso



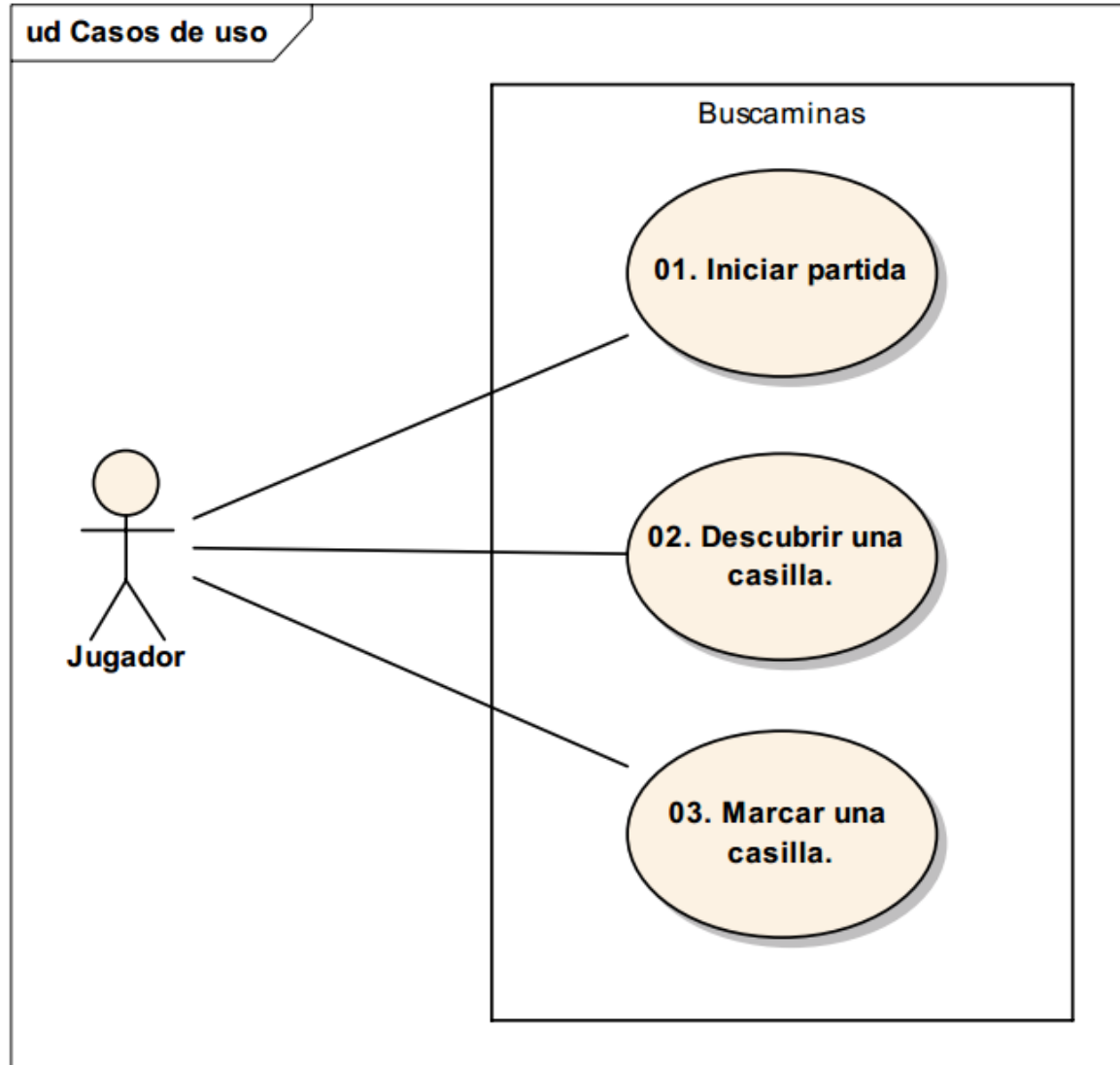
¿Qué casos de uso identificamos?

- » Iniciar una nueva partida.
- » Descubrir una casilla.
- » Marcar una casilla.

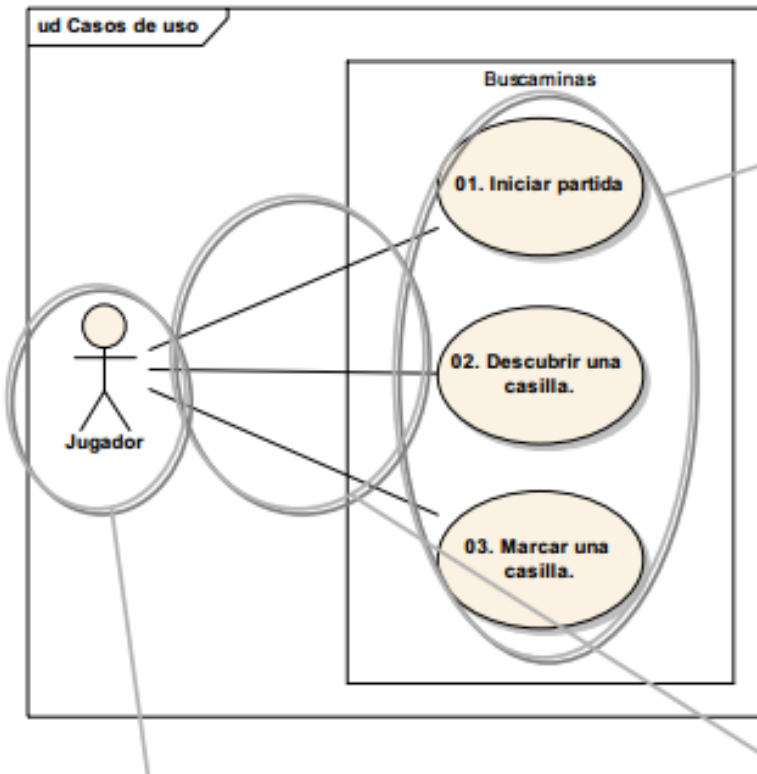
¿Quién realiza estos casos de uso?

- » El jugador.

# Diagramas de casos de uso



# Diagramas de casos de uso



**Caso de Uso:** interacción entre actores y el sistema que produce un resultado observable de valor para un actor.

**Límite del sistema:** agrupa casos de uso dentro de un mismo sistema. Útil cuando tenemos varios sistemas / subsistemas.

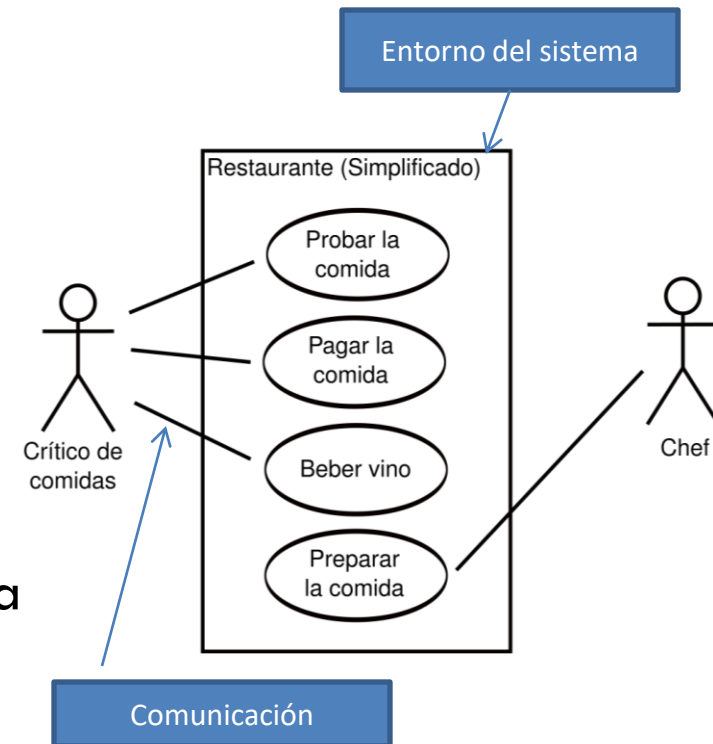
**Actor:** alguien o algo externo al sistema que interactúa con él desempeñando un rol.

Un caso de uso **siempre** es iniciado por un actor externo.

**Asociación:** la participación de un actor es necesaria para realizar el caso de uso.

# Diagramas de casos de uso. Elementos

- ❑ **Actores:** representan un *tipo de usuario* del sistema, es decir, cualquier cosa que interactúa con el sistema y es externo a él (puede ser un humano, otro sistema informático, un dispositivo, una empresa, unidades organizativas ... )  
Se representan mediante un **monigote** con un nombre debajo (comenzando en mayúscula).
- ❑ **Caso de uso:** representa una *unidad funcional* del sistema que se realiza tras una orden de algún agente externo que puede ser un actor u otro caso de uso.  
Se representan por una **elipse** en cuyo interior se escribe una descripción.
- ❑ **Relaciones:** normalmente una línea continua que representa la interacción o asociación entre actores y casos de uso. Existen diferentes tipos.
- ❑ **Limite del sistema:** rectángulo con el entorno del sistema.



# Actores

- Representan **roles** que personas o elementos desempeñan en el sistema.
- Se les da un nombre que describa el papel que desempeñan.
- Los actores son siempre externos al sistema e interactúan directamente con él.
- Un mismo usuario puede tener **diferentes roles** → **actores distintos**.
- Pueden ser un humano, otro sistema, unidades organizativas, una empresa ...
- **Tipos de actores:**
  - **Primarios o activos:** activan el caso de uso.
  - **Secundarios:** interactúan con el caso de uso después de haberse activado, suelen ser otros sistemas, componentes externos, etc.
  - **Iniciadores:** no interactúan con el sistema pero desencadenan el trabajo de otro actor.



# Caso de uso

---

- Un caso de uso:
  - Secuencia de acciones realizadas por el sistema.
  - Producen un resultado observable.
  - Participan actores.

# Caso de uso

---

- ❑ El objetivo principal no es crear el diagrama en sí, sino la descripción que de cada caso de uso se hace, ya que esto ayuda al desarrolladora crear el sistema.

# Identificar casos de uso

- ❑ Para **identificar** los casos de uso formularemos una serie de preguntas:
  - ❑ ¿Qué tareas realizan los actores del sistema?
  - ❑ ¿Qué información crea, almacena, modifica, destruye, lee el actor?
  - ❑ ¿Debe el actor notificar al sistema los cambios externos ocurridos?
  - ❑ ¿Debe el sistema informar al actor de cambios internos?
  - ❑ ¿Interactúa el sistema con algún sistema externo?

# Descripción del caso de uso: contrato

- ❑ El modelado de casos de uso consiste en escribir texto.
- ❑ Son documentos de texto, no son diagramas.
- ❑ Describimos el flujo de eventos:
  - ❑ Texto informal
  - ❑ Texto estructurado formal (plantillas)
- ❑ Podemos hacerlo en tablas con un editor de texto habitual y con ArgoUML también podremos hacerlo en la pestaña de documentación.
- ❑ Debe ser legible y comprensible para el usuario no experto.
- ❑ Debe indicar: actores, flujos principal y excepciones.

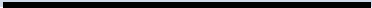


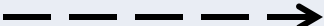
# Descripción del caso de uso: contrato

- ❑ Una **plantilla para la especificación de un caso de uso** puede ser:
  - ✓ **Nombre:** nombre del caso de uso.
  - ✓ **ID:** id del caso de uso.
  - ✓ **Propósito:** Breve descripción de lo que se espera que haga el caso de uso.
  - ✓ **Actores:** actores implicados en el caso de uso.
  - ✓ **Precondiciones:** condiciones que deben cumplirse para que pueda llevarse a cabo el caso de uso.
  - ✓ **Flujo normal:** *pasos o eventos que deben cumplirse para ejecutar el caso de uso exitosamente, desde el punto de vista del actor que participa y del sistema. Los pasos se ordenan cronológicamente.*
  - ✓ **Flujo alternativo:** pasos o eventos que se llevan a cabo cuando se producen casos inesperados o poco frecuentes. Por ejemplo, un usuario accede a un sistema escribiendo su nombre de usuario y contraseña, y los escribe mal -> el sistema debe informar de esta situación.
  - ✓ **Postcondiciones:** condiciones que se cumplen una vez que se ha realizado el caso de uso.

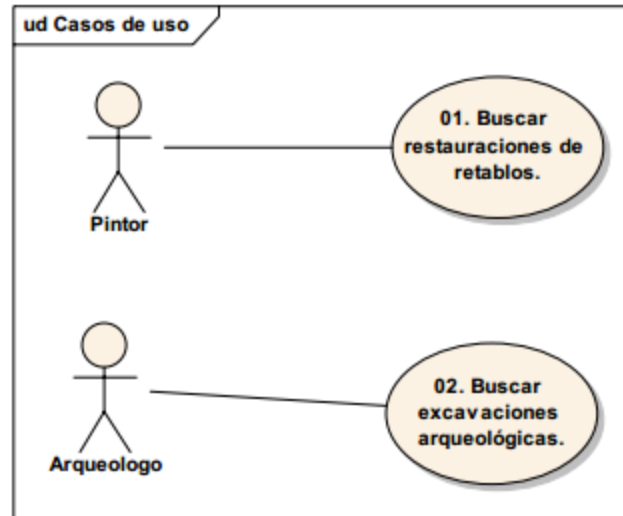
# Escenarios.

- ❑ Un **escenario** es una **ejecución particular o instancia** de un caso de uso que se describe como una secuencia de eventos.
- ❑ Cada escenario acaba con éxito o fracaso.
- ❑ El caso de uso consta de un flujo principal y varios flujos secundarios.
  - ❑ Flujo principal: “todo va bien”
  - ❑ Flujos secundarios: alternativas y excepciones.
- ❑ **Un caso de uso es una colección de escenarios (generalización de un escenario)**
- ❑ **Un escenario es una secuencia particular de pasos.**
- ❑ Los escenarios pueden y deben posteriormente documentarse mediante **diagramas de secuencia.**

# Relaciones

RELACIÓN	FUNCIÓN	NOTACIÓN
<b>Asociación o Interacción</b>	Línea de comunicación <b>entre un actor y un caso de uso</b> en el que participa.	
<b>Extensión</b> <b>&lt;&lt;extend&gt;&gt;</b>	Relación que se puede dar <b>entre casos de uso</b> . Permite que un caso de uso extienda su comportamiento. Se utiliza para representar relaciones entre un caso de uso que requiere la ejecución de otro en determinadas circunstancias. La flecha apunta al caso de uso que se extenderá.	<b>&lt;&lt;extend&gt;&gt;</b> 
<b>Generalización</b>	<ul style="list-style-type: none"><li>• Se utiliza para representar relaciones de herencia <b>entre casos de uso</b>. El caso de uso hijo hereda el comportamiento y significado del padre. El caso de uso padre se define de forma abstracta. Se usa menos que en los diagramas de clases.</li><li>• También se utiliza para representar la relación de herencia <b>entre actores</b>.</li></ul>	
<b>Inclusión</b> <b>&lt;&lt;include&gt;&gt;</b> o <b>&lt;&lt;uses&gt;&gt;</b>	Relación que se puede dar <b>entre casos de uso</b> . Permite que un caso de uso base incluya el comportamiento de otro caso de uso. Se utiliza cuando queremos dividir una tarea de mayor envergadura en otras más sencillas, que son utilizadas por la primera. La flecha apunta al caso de uso incluido.	<b>&lt;&lt;include&gt;&gt;</b> 

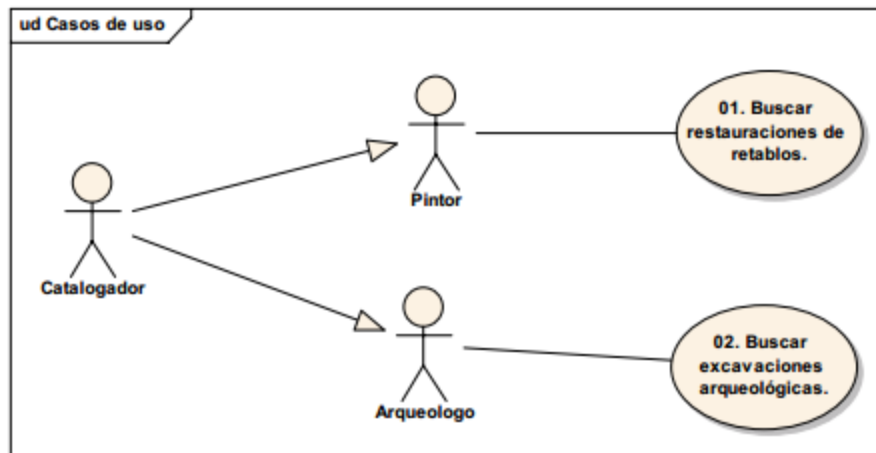
# Relaciones: generalización entre actores



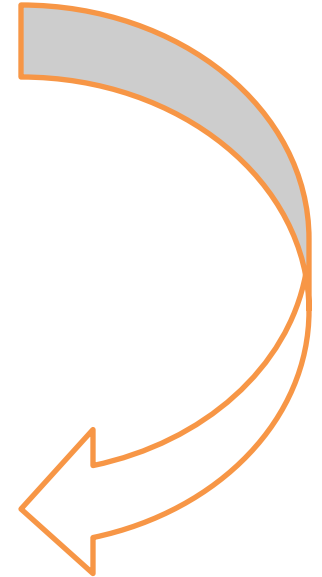
Deseamos un tercer actor *catalogador* cuya misión sea catalogar *retablos* y *excavaciones* de la misma manera que un pintor o arqueólogo..

Alternativas:

1. Repetir los casos de uso para el actor *catalogador*.
  2. Añadir al actor *catalogador*
- Etc...



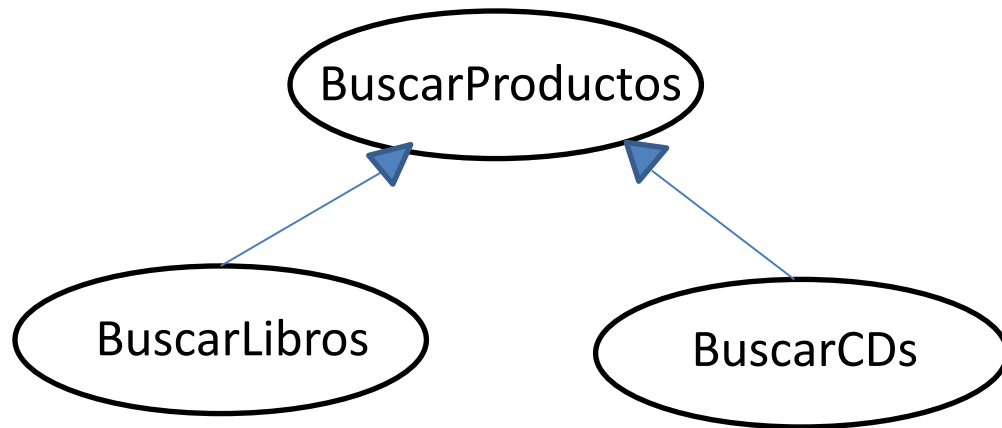
Definir al actor *catalogador* como una extensión de los actores *pintor* y *arqueólogo*.



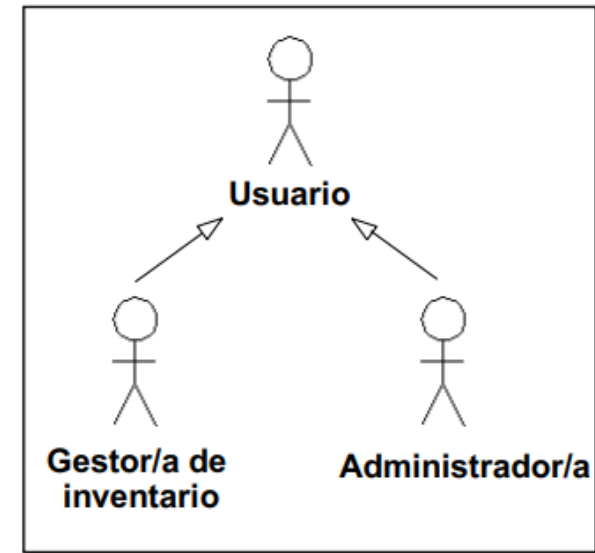


# Relaciones: generalización entre casos de uso

- ❑ La relación de generalización se utiliza cuando se tiene uno o más casos de uso que son una especificación de un caso de uso más general. También se puede establecer esta relación entre actores.



Por ejemplo, supongamos un sistema de ventas en el que se pueda buscar dos tipos de productos, libros y CD's. Podríamos tener un caso de uso BuscarProducto, y dos especificaciones BuscarLibros y BuscarCD (ambos son refinamientos del caso de uso padre).

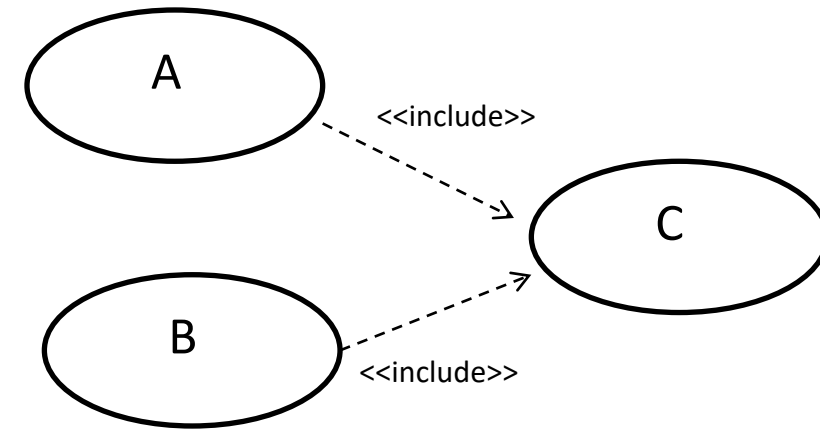


Relaciones de generalización entre actores

Los actores hijos pueden jugar todos los roles del actor padre

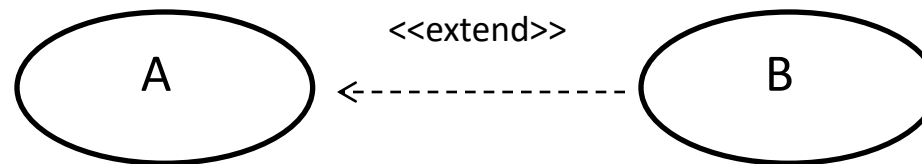
# Relaciones: inclusión entre casos de uso

- ❑ La relación `<<include>>` ocurre cuando se tiene una porción de comportamiento que es similar en más de un caso de uso. Si por ejemplo, tenemos dos casos de uso A y B que tienen una serie de pasos en común, se ponen en un tercer caso de uso C; y A y B lo incluyen para usarlo. Es necesario que ocurra el caso de uso incluido para satisfacer el objetivo del caso de uso base.
- ❑ **Ventajas:**
  - ❑ Las descripciones de los casos de uso son más cortas
- ❑ **Desventajas:**
  - ❑ La lectura de los diagramas es más difícil, sobre todo para los clientes.



# Relaciones: extensión entre casos de uso

- ❑ La relación <<extend>> se usa cuando un caso de uso extiende o amplía la funcionalidad de otro (caso de uso base). Por ejemplo, el caso de uso B extiende la funcionalidad de A añadiendo algunos pasos.
- ❑ Se diferencia del <<include>> en que el caso de uso que es extendido no sabe nada del caso de uso lo extiende.
- ❑ B no es indispensable que ocurra, ni va a ocurrir siempre, pero cuando ocurre ofrece un valor extra al objetivo de A.



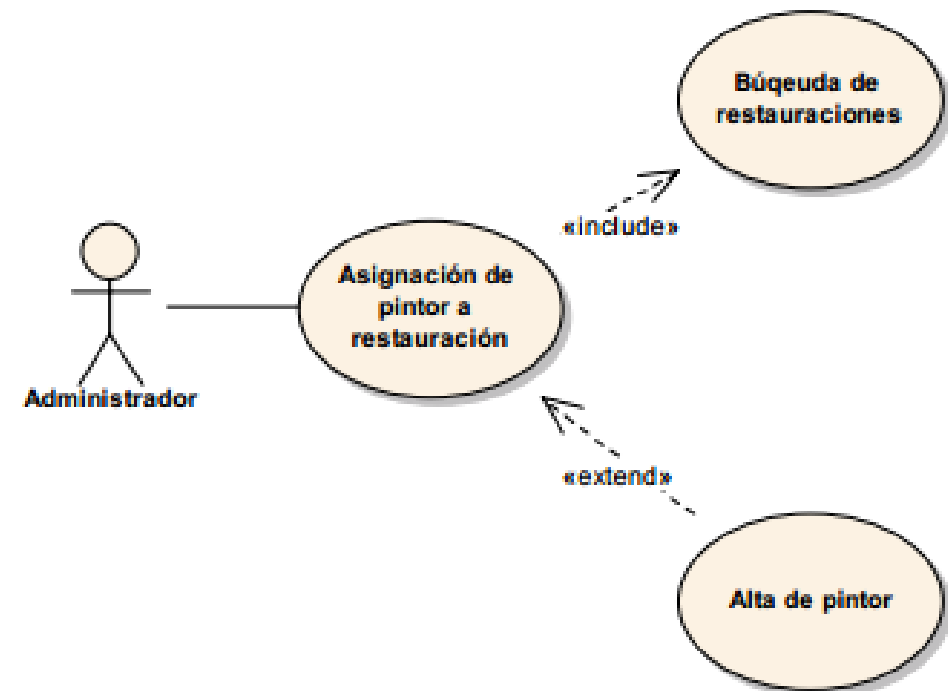
# Relaciones: extensión vs inclusión

Un *actor* administrador puede entrar en el sistema, **asignar a un pintor una restauración** y marcharse.

Para elegir una restauración a la que asignar un pinto, el administrador **debe** realizar una búsqueda entre todas las restauraciones existentes y seleccionar una.

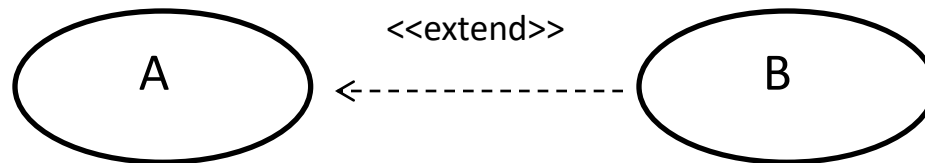
## Inclusión

ud Ejemplos de casos de uso



# Estereotipo.

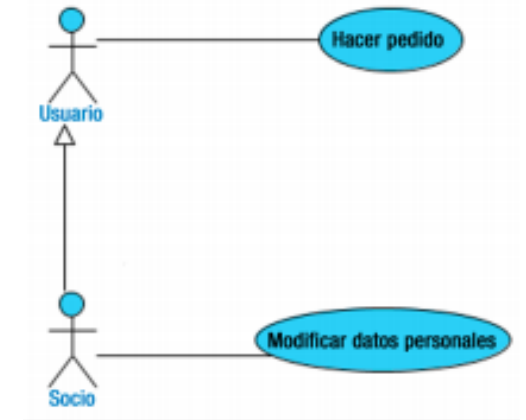
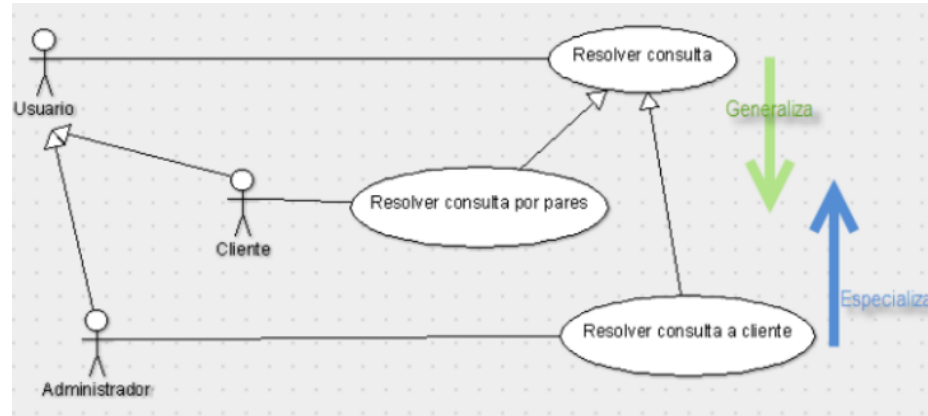
- ❑ Elemento de texto que al ser aplicado a otro elemento define su categoría.
- ❑ Pequeñas etiquetas de texto que aplicadas a otros elementos o relaciones aportan significado adicional.
- ❑ Se escriben entre << >>



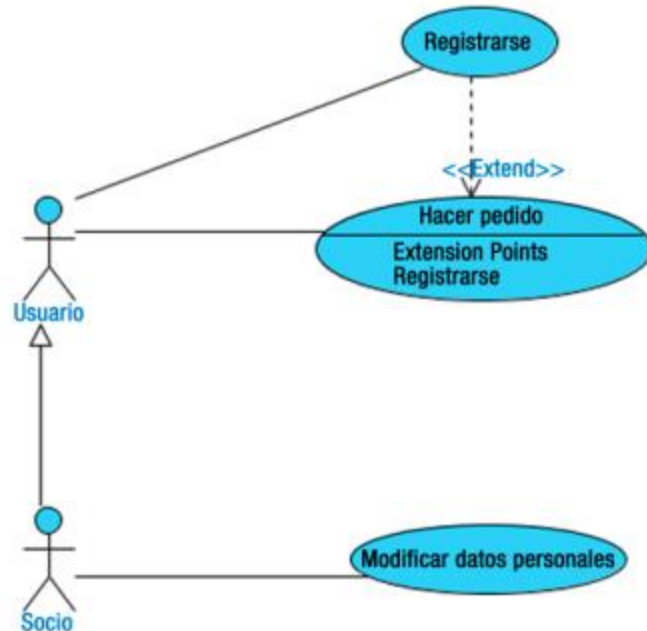
# Ejemplos Relaciones.



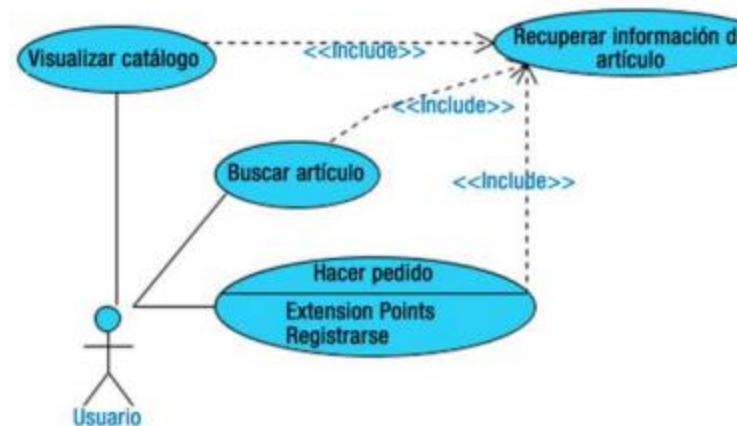
Asociación



Generalización

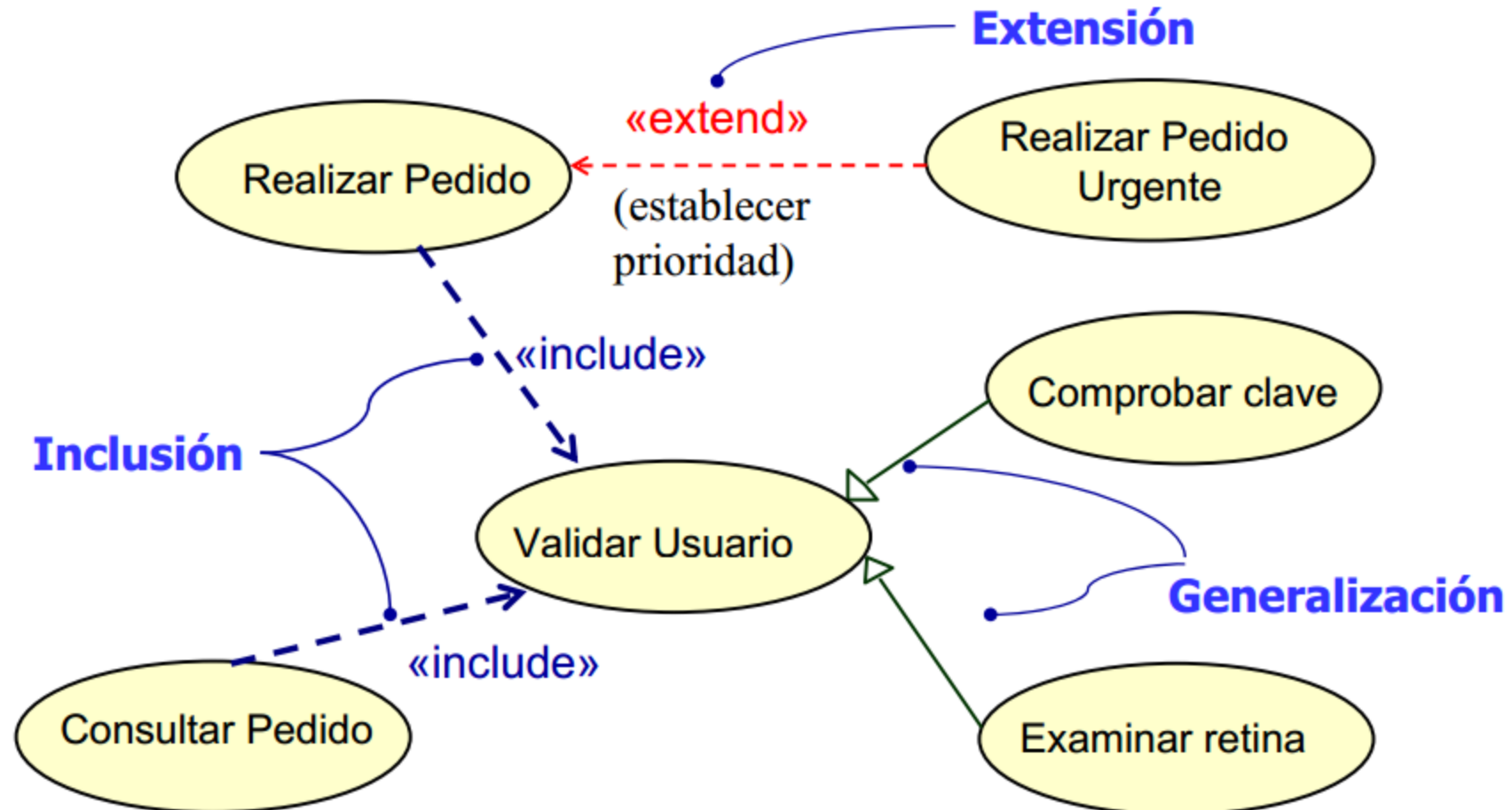


Extensión



Inclusión

# Ejemplos Relaciones.



# Elaboración.

- ❑ Partiremos de una descripción lo más detallada posible del problema a resolver y trataremos de detectar quien interactúa con el sistema, para obtener los actores diagrama de casos de uso, a continuación buscaremos qué tareas realizan estos actores para determinar los casos de uso más genéricos. El siguiente paso es refinar el diagrama analizando los casos de uso más generales para detectar casos relacionados por inclusión (se detectan fácilmente cuando aparecen en dos o más casos de uso generales), extensión y generalización. Al diagrama generado se le denomina diagrama frontera.



# Elaboración.

---

- 1) Identificar los **usuarios** del sistema.
- 2) Encontrar todos los **roles** que juegan los usuarios y que son relevantes al sistema.
- 3) Para cada rol identificar todas las formas (**objetivos**) de interactuar con el sistema.
- 4) **Crea un caso de uso por cada objetivo.**
- 5) Estructurar los casos de uso. (**¡Cuidado!**)
- 6) Revisar y **validar** con el usuario.

# Elaboración.

- ❑ La construcción del diagrama de caso de uso consta de:
  - ❑ Diagrama de contexto y Diagrama inicial o diagrama frontera.
  - ❑ Plantillas de descripción de los casos de uso
  - ❑ Diagrama Estructurado o modelo de casos de uso.

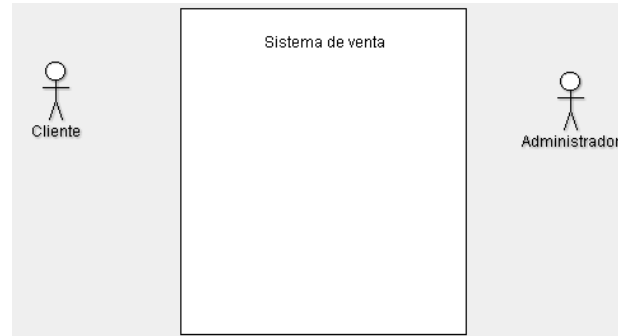


Diagrama contexto

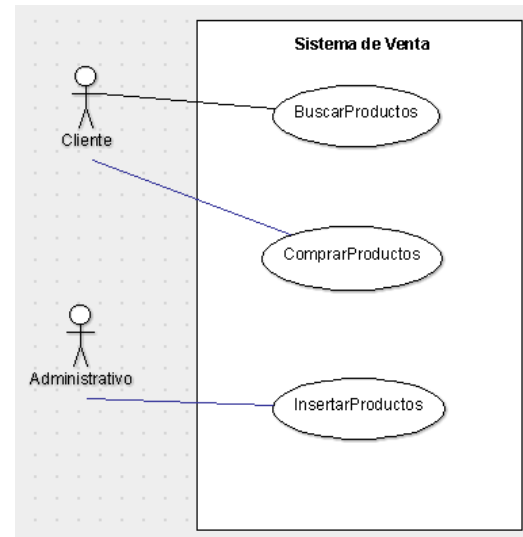


Diagrama inicial o frontera

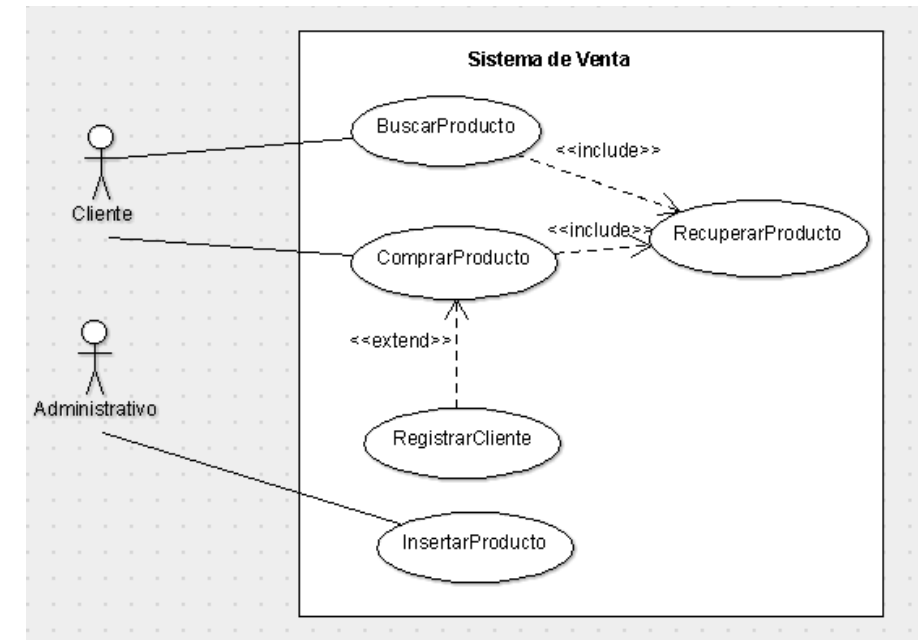


Diagrama Estructurado

# Elaboración.

- ❑ Paso 1: identificamos el Diagrama de contexto: se le da un nombre e identifica el entorno del sistema, es decir, los límites del sistema software.
- ❑ Paso 2: identificamos los actores que interactúan con el sistema.
- ❑ Paso 3: Refinamos el *diagrama de contexto* y obtenemos el *diagrama inicial*, en el cual identificamos los casos de uso o funcionalidades del sistema.
- ❑ Paso 4: Refinamos más el diagrama: obtenemos las relaciones de inclusión, extensión, generalización y obtenemos así el *diagrama estructurado* o *modelo de casos de uso*.

# Diagramas de casos de uso. Ejemplos

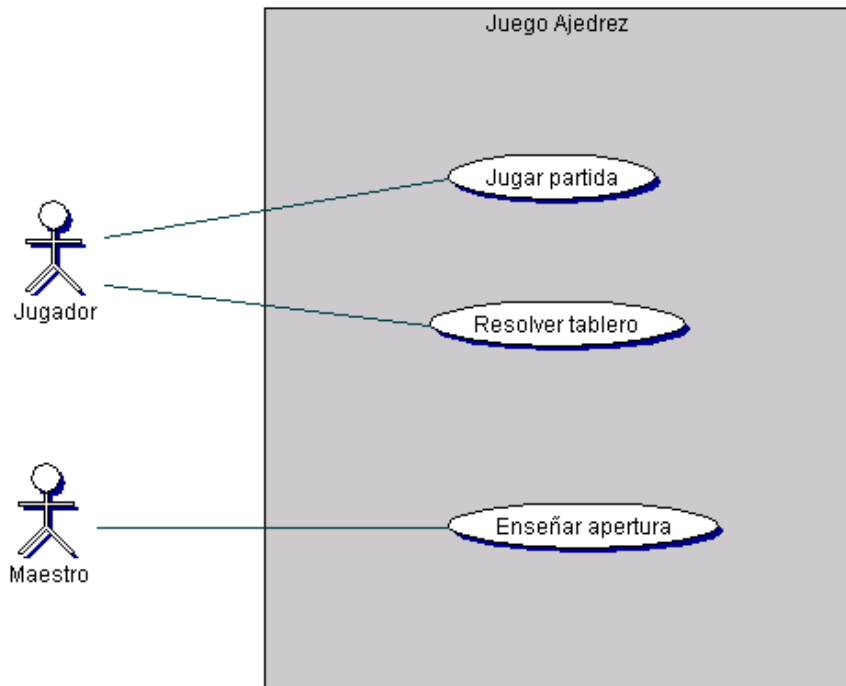


Diagrama que representa dos actores interactuando con un sistema para jugar al ajedrez.

En el diagrama se han reflejado 3 casos de uso o funcionalidades que permite el sistema.

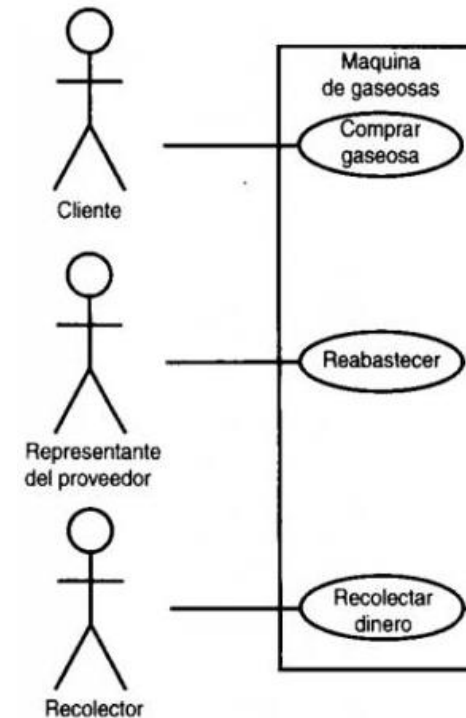
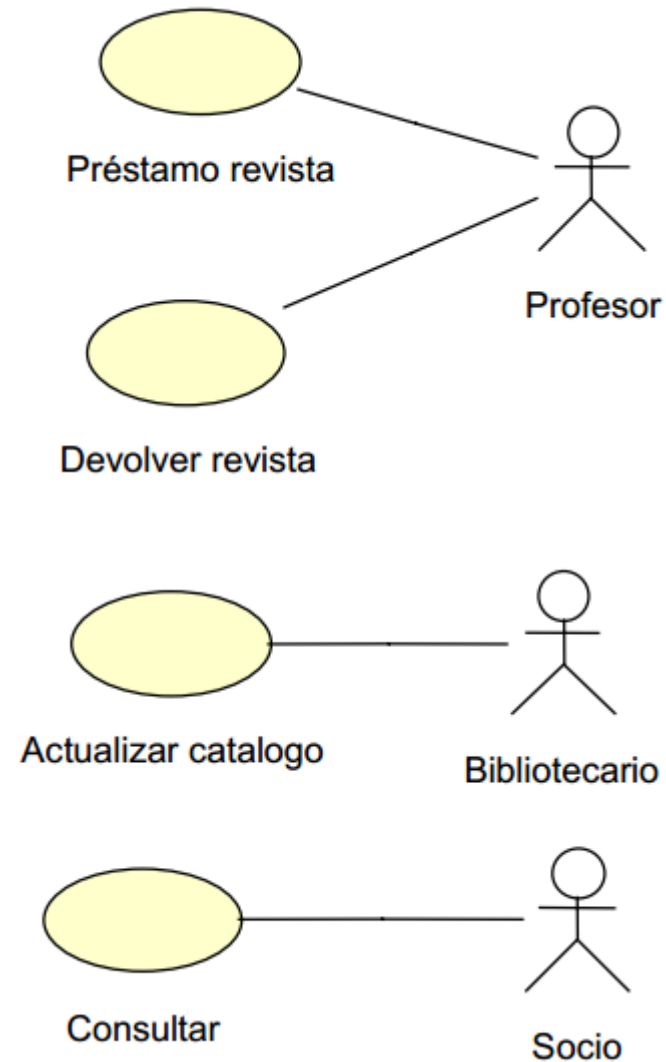
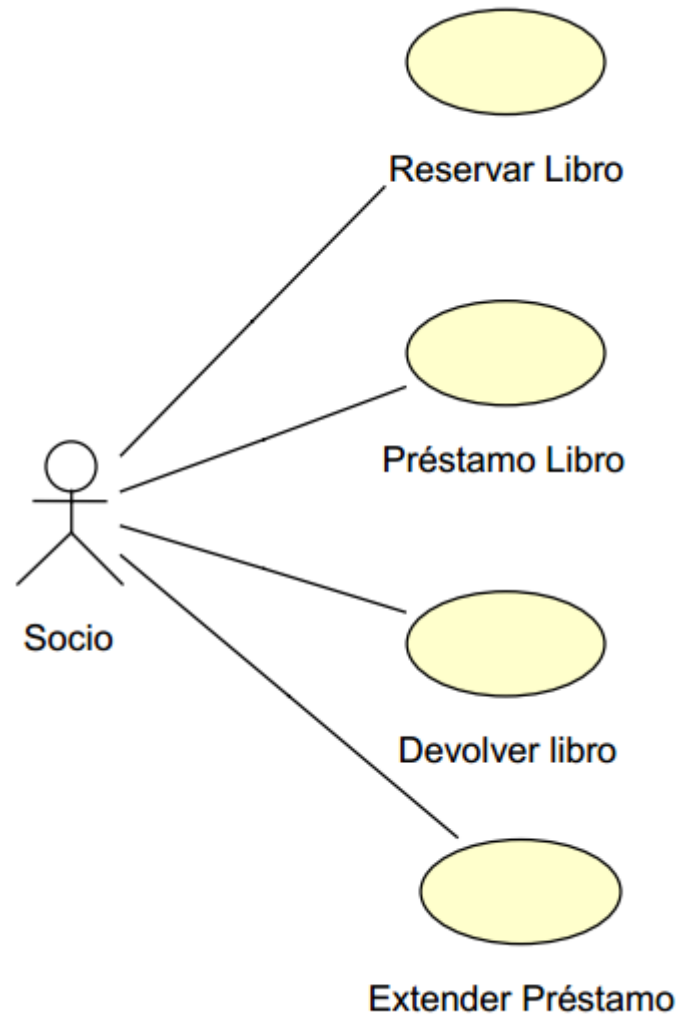


Diagrama que representa 3 actores interactuando con un sistema de venta de refrescos. El cliente interactúa con el sistema para buscar y comprar un producto. El proveedor inserta nuevos productos y el recolector recoge el dinero.

En el diagrama se han reflejado 3 casos de uso o funcionalidades que permite el sistema.

# Diagramas de casos de uso. Ejemplos

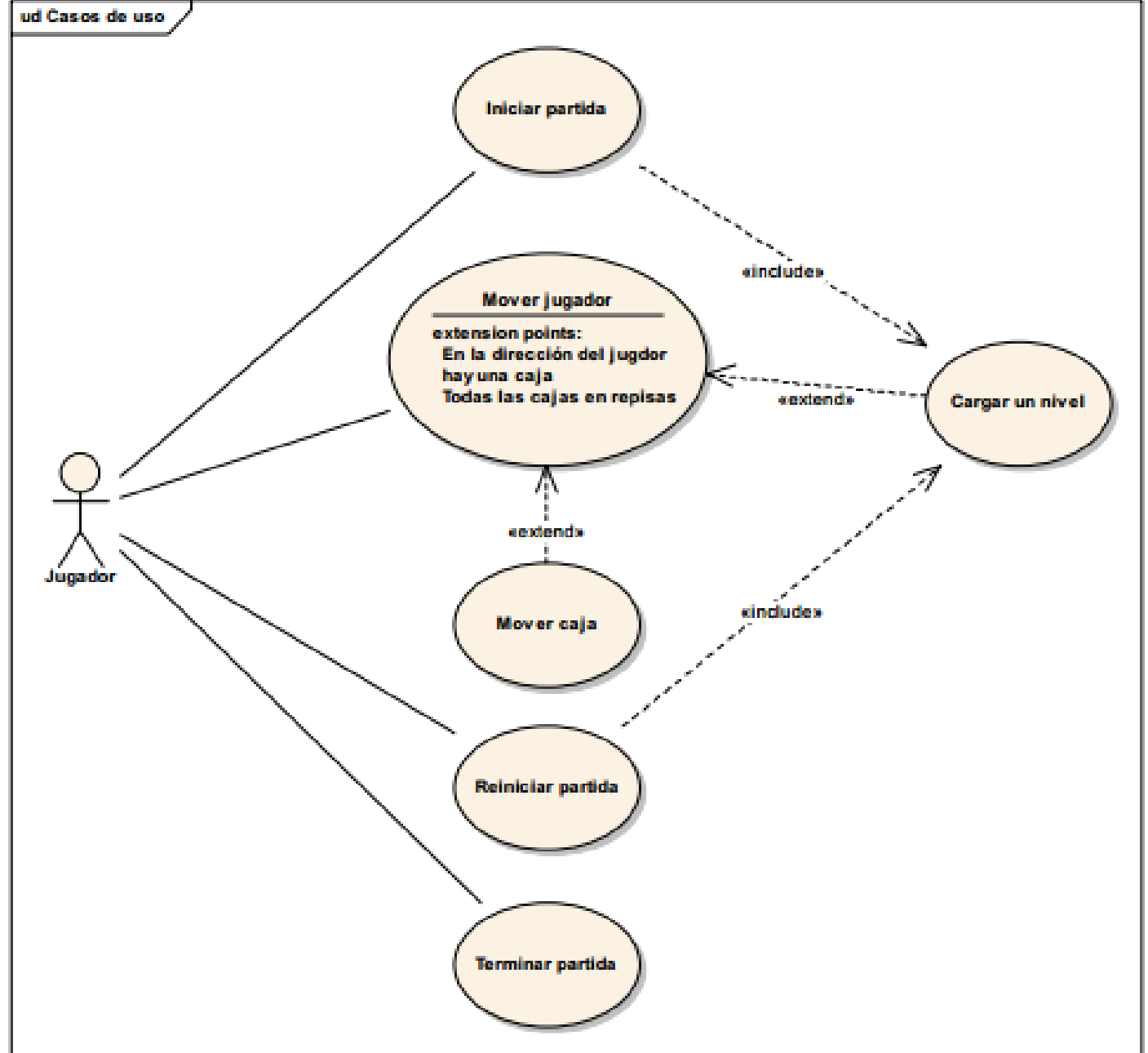


## Ejemplo

- ❑ Sokoban es un juego de varios niveles.
- ❑ Cada nivel está compuesto por un jugador, cajas, repisas y muros.
- ❑ El objetivo del jugador es empujar todas las cajas sobre las repisas.
- ❑ Cuando esto sucede el jugador pasa al siguiente nivel.
- ❑ Para mover una caja, el jugador debe colocarse al lado y empujarla. Si la casilla hacia la que está empujando la caja está libre la caja se moverá.
- ❑ Si el jugador se queda bloqueado, es decir, no puede terminar el nivel, puede reiniciar el nivel perdiendo una vida.
- ❑ Cuando el jugador pierde todas sus vidas la partida termina.



# Ejemplo



# Ejemplo de caso de uso

## **Realizar Venta** (en un terminal de punto de venta, TPV)

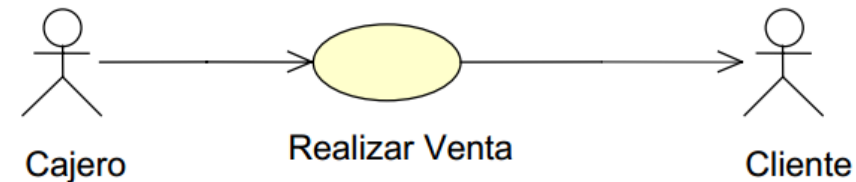
**Actor :** Cajero

### **Descripción:**

Un cliente llega al TPV con un conjunto de artículos. El Cajero registra los artículos y se genera un ticket. El cliente paga en efectivo y recoge los artículos.

### **Flujo:**

1. El cliente llega al TPV con los artículos.
2. El cajero registra el identificador de cada artículo.
3. El sistema obtiene el precio de cada artículo y añade la información a la transacción de venta.
4. Al acabar el cajero indica la finalización de la introducción de artículos.
5. El sistema calcula el total de la compra y lo muestra.





# Ejemplo de caso de uso

## **Realizar Venta** (en un terminal de punto de venta, TPV)

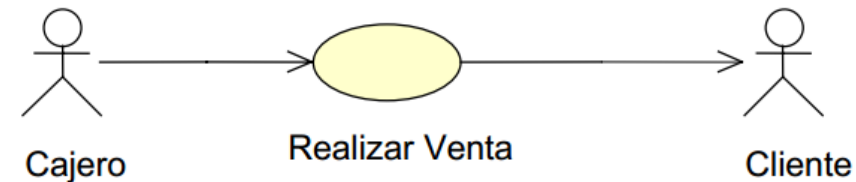
**Actor :** Cajero

### **Descripción:**

Un cliente llega al TPV con un conjunto de artículos. El Cajero registra los artículos y se genera un ticket. El cliente paga en efectivo y recoge los artículos.

### **Flujo:**

1. El cliente llega al TPV con los artículos.
2. El cajero registra el identificador de cada artículo.
3. El sistema obtiene el precio de cada artículo y añade la información a la transacción de venta.
4. Al acabar el cajero indica la finalización de la introducción de artículos.
5. El sistema calcula el total de la compra y lo muestra.



# Ejemplo de caso de uso: Hacer pedido

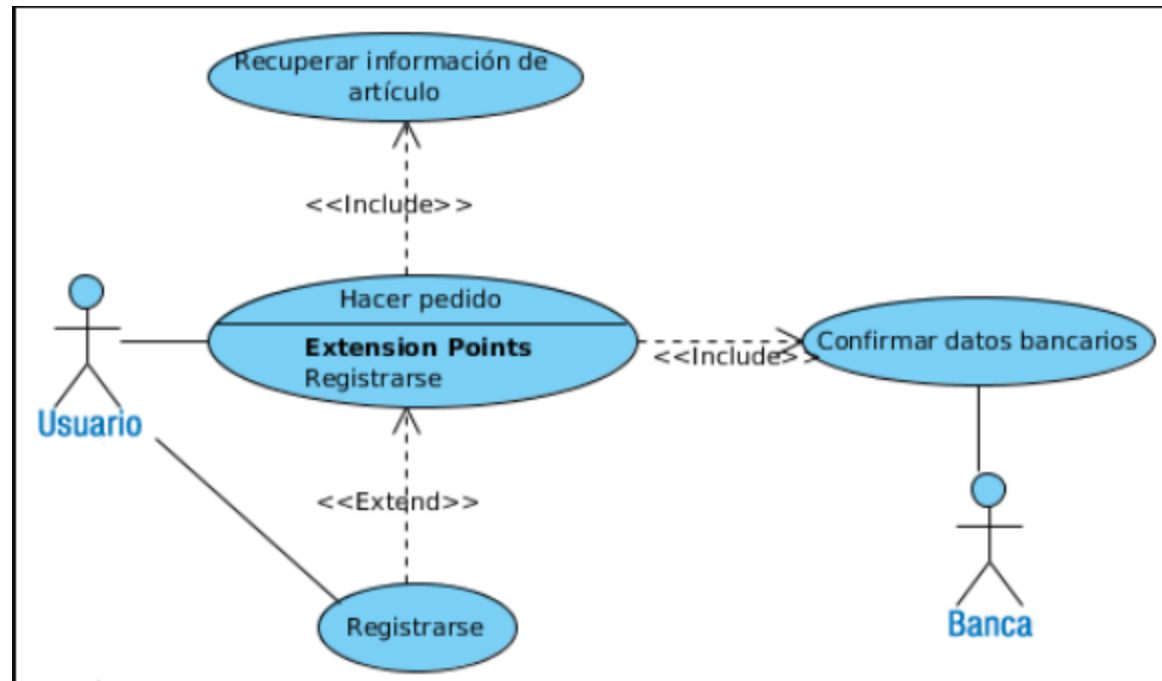
Los usuarios del sistema navegan por la web para ver los artículos, zapatos, bolsos y complementos que se venden en la tienda.

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

Los socios **pueden hacer pedidos de los artículos**. 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 cuanto debe pagar el socio en total. El pago se hace a través tarjeta bancaria, cuando se va a pagar una entidad bancaria comprueba la validez de la tarjeta. De la tarjeta interesa conocer el número.

....

# Ejemplo de caso de uso: Hacer pedido



# Ejemplo de caso de uso: Hacer pedido

**Nombre:** HacerPedido

**ID:** CU-1

**Descripción:**  
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.

**Actores:** Cliente

**Precondiciones:** Existe un catálogo de productos disponibles para pedir. El usuario está registrado. Los datos bancarios son correctos.

**Flujo normal:**

1. El cliente selecciona hacer pedido.
2. El sistema crea un pedido en construcción.
3. El cliente selecciona un articulo.
4. El sistema recupera la información del articulo para obtener precio.
5. El cliente selecciona la cantidad.
6. El sistema calcula el total del pedido.
7. Se repite los pasos 3 a 6 hasta completar la lista de artículos.
8. El cliente solicita pagar.
9. El sistema comprueba que el cliente está registrado, si no muestra un aviso para que se registre.
10. El sistema calcula y añade al importe los gastos de envío.
11. El sistema solicita una forma de pago.
12. El usuario indica la forma de pago.
13. El sistema solicita los datos asociados a la forma de pago y los comprueba.
14. Se realiza el pago y el pedido se registra en estado “pendiente”.

**Postcondiciones:** Se crea un pedido con los datos del usuario que lo realiza y los artículos solicitados.

**Flujo alternativo 1:**

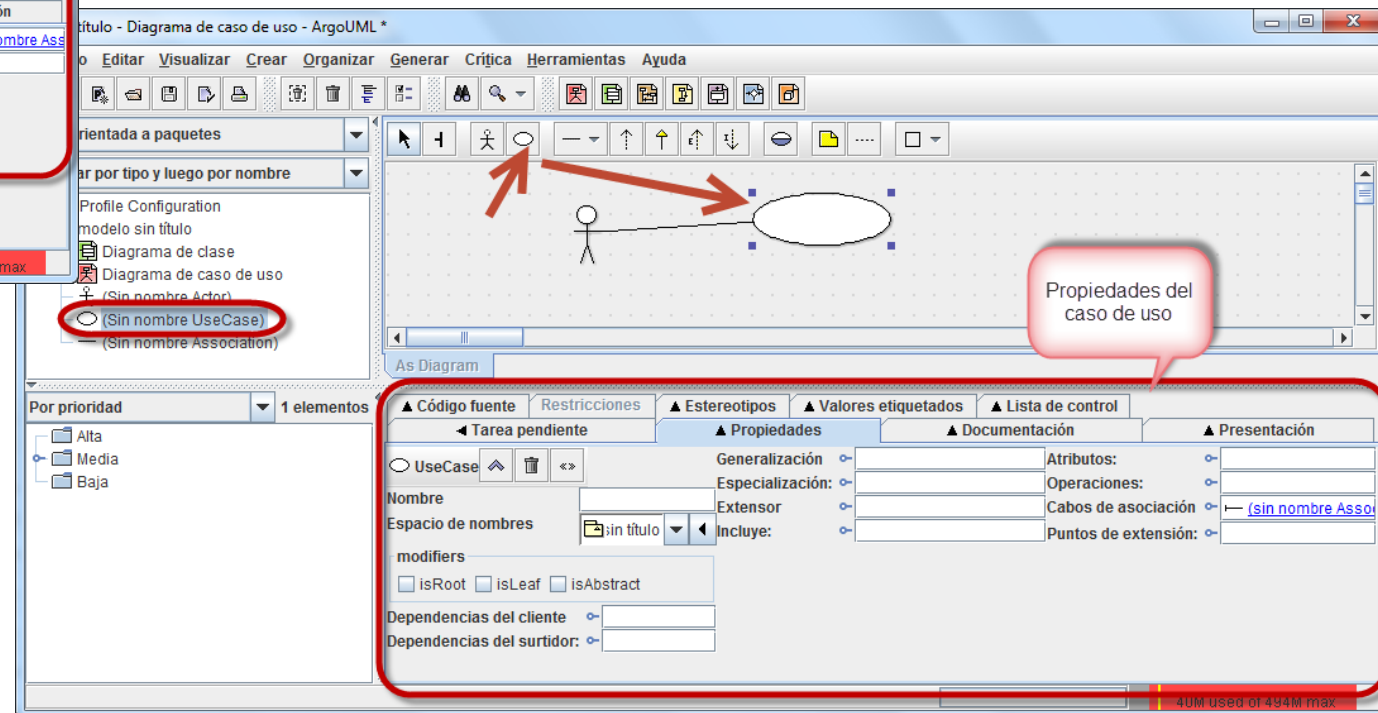
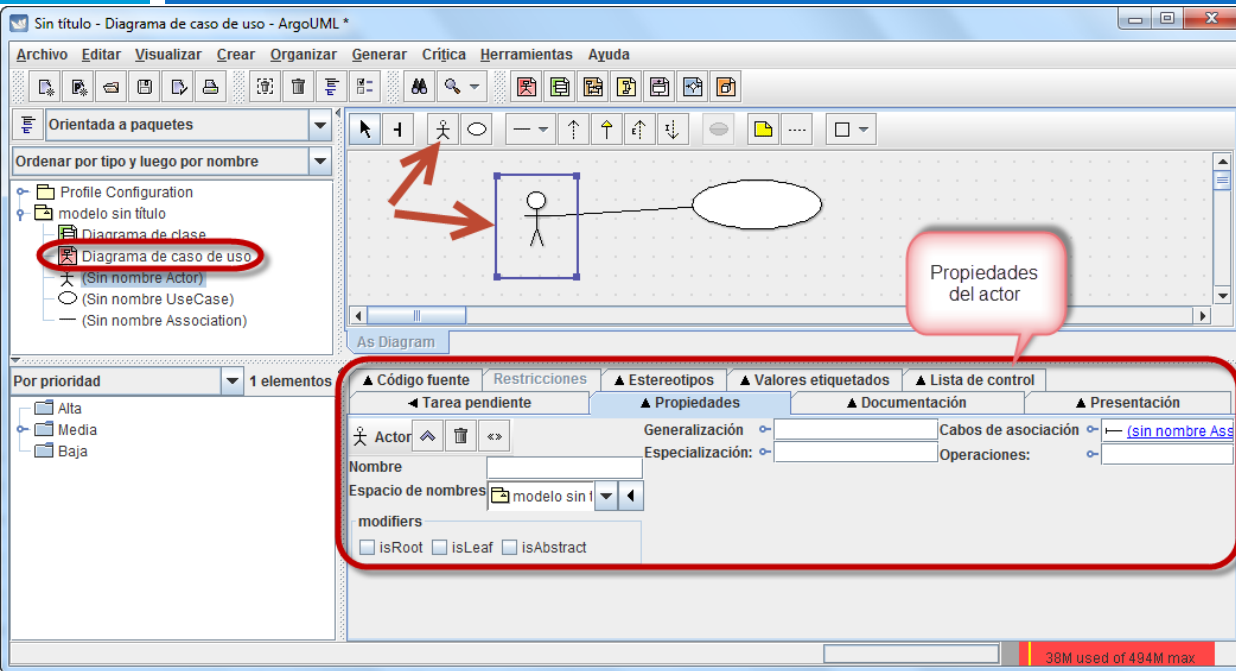
- 9.1. El sistema invoca el registro de usuario.
- 9.2. El sistema solicita los datos personales.
- 9.3. El cliente introduce los datos personales.
- 9.4. Fin del caso de uso.

...

# Ejemplo de escenario.

- ❑ **Ejemplo:** El caso de uso “Realizar pedido” → Tendrá varios escenarios. → Un escenario puede ser: “Realizar pedido de unos zapatos”
  - ❑ El usuario inicia el pedido.
  - ❑ Se crea el pedido en estado “en construcción”
  - ❑ Se selecciona un par de zapatos “Lucía” de piel negros, número 38.
  - ❑ Se selecciona la cantidad 1
  - ❑ Se recupera la información de los zapatos y se modifica la cantidad a pagar 45€
  - ❑ El usuario acepta el pedido.
  - ❑ Se comprueba si el usuario está registrado.
  - ❑ Se comprueban los datos bancarios.
  - ❑ Se calcula el total a pagar añadiendo los gastos de envío.
  - ❑ Se realiza el pago a través de una entidad externa.
  - ❑ Se genera el pedido para el usuario con estado “pendiente”.

# Diagramas de casos de uso. ArgoUML



# Buenas prácticas.

- ❑ Los casos de uso deben tener una redacción cuidada para evitar problemas de interpretación.
- ❑ El caso de uso debe describir qué debe hacer el sistema a desarrollar en su interacción con los actores y no cómo debe hacerlo. Es decir, debe describir sólo comportamiento observable externamente, sin entrar en la funcionalidad interna del sistema.
- ❑ El nombre del caso de uso debe ilustrar el objetivo que pretende alcanzar el actor al realizarlo.
- ❑ El caso de uso debe describir interacciones con los actores sin hacer referencias explícitas a elementos concretos de la interfaz de usuario del sistema a desarrollar.
- ❑ La invocación de unos casos de uso desde otros casos de uso (inclusión, o extensión), sólo debe usarse como un mecanismo para evitar repetir una determinada secuencia de pasos que se repite en varios casos de uso. Nunca debe usarse para expresar posibles menús de la interfaz de usuario.
- ❑ Se debe ser cuidadoso al usar estructuras condicionales en la descripción del caso de uso, ya que los clientes y usuarios no suelen estar familiarizados con este tipo de estructuras, especialmente si son complejas.
- ❑ Se debe intentar que todos los casos de uso de una misma especificación de requisitos de software (ERS) estén descritos al mismo nivel de detalle.
- ❑ En los diagramas de casos de uso, debe evitarse que se crucen las líneas que unen los actores a los casos de uso.



### 3. Diagramas de interacción.



## 3.1 Diagramas de Secuencia.

# Diagramas de secuencia.

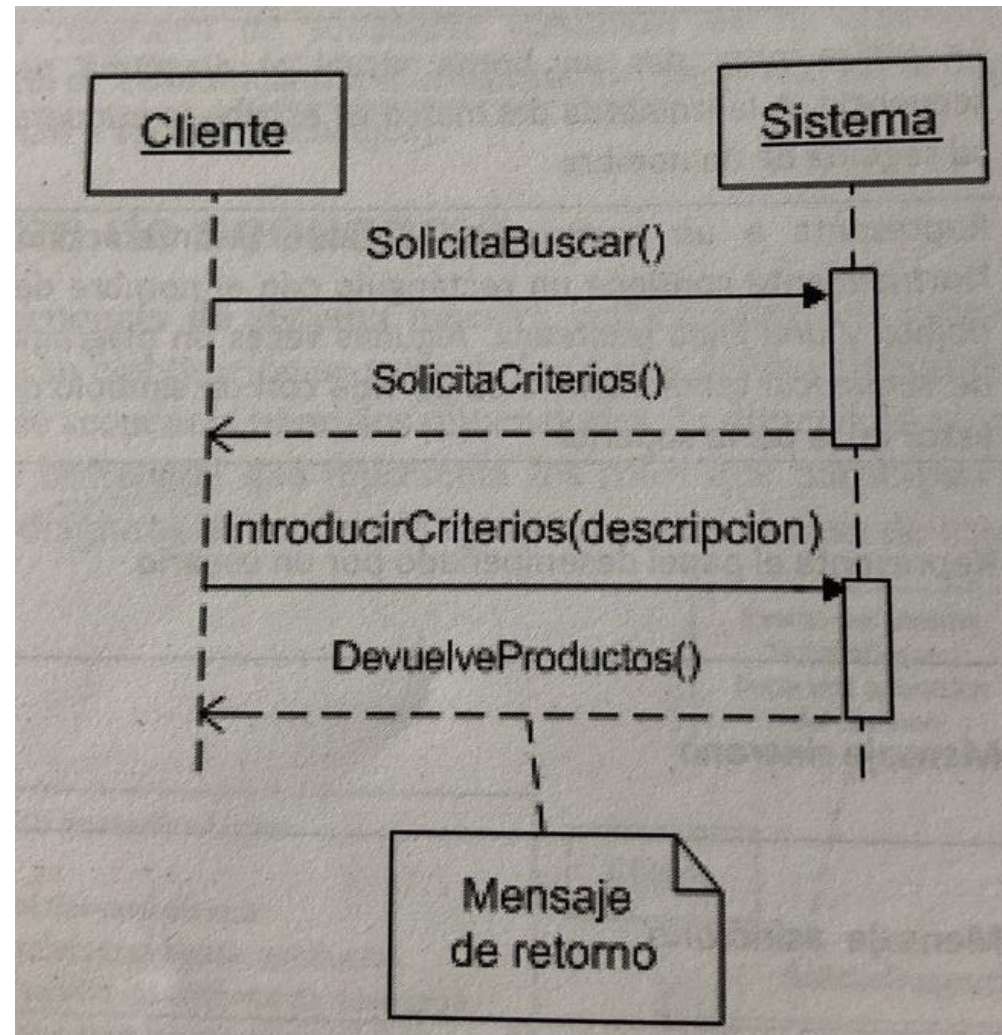
- ❑ Muestran la **secuencia temporal** de los mensajes que intercambian los actores u objetos del sistema.
- ❑ Se modela para cada caso de uso y teniendo en cuenta el diagrama de clases.
  - ❑ Al tratarse de una comunicación, se debe identificar el emisor y receptor de dicha comunicación.
  - ❑ Los mensajes se ordenan en el tiempo (el orden es importante).
- ❑ Constan de 2 dimensiones:
  - ❑ **Vertical:**
    - ❑ Se define una línea temporal vertical (**línea de vida**) para cada actor u objeto en la que se representan los mensajes que intercambian.
  - ❑ **Horizontal:** se alinean en la parte superior los roles u objetos que participan en la interacción.
    - ❑ Cada actor u objeto se representa mediante un rectángulo distribuido horizontalmente en la zona superior del diagrama. A cada uno se le asocia una línea vertical llamada **línea de vida**.
  - ❑ Los **mensajes** (invocación a métodos) representan la comunicación entre los participantes . Se representan mediante flechas horizontales de una línea de vida a otra, indicando la flecha la dirección del mensaje y el orden.

# Diagramas de secuencia.

Caso de uso: BuscarProducto:

El primer mensaje (evento) inicia la operación de búsqueda de productos en el sistema.

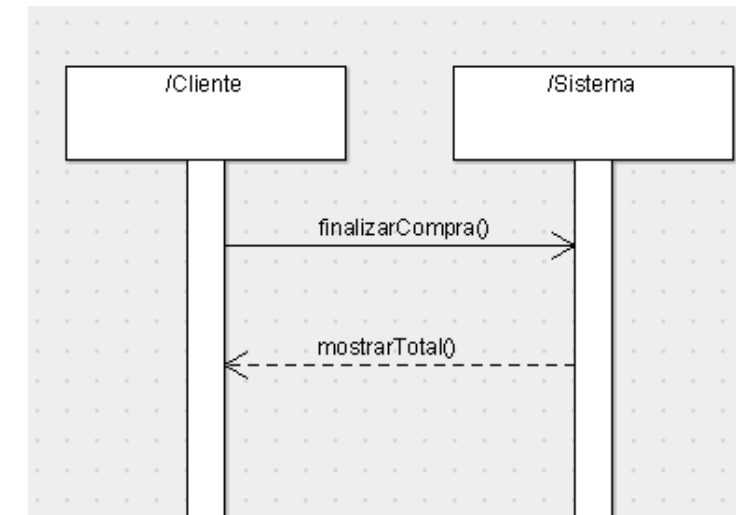
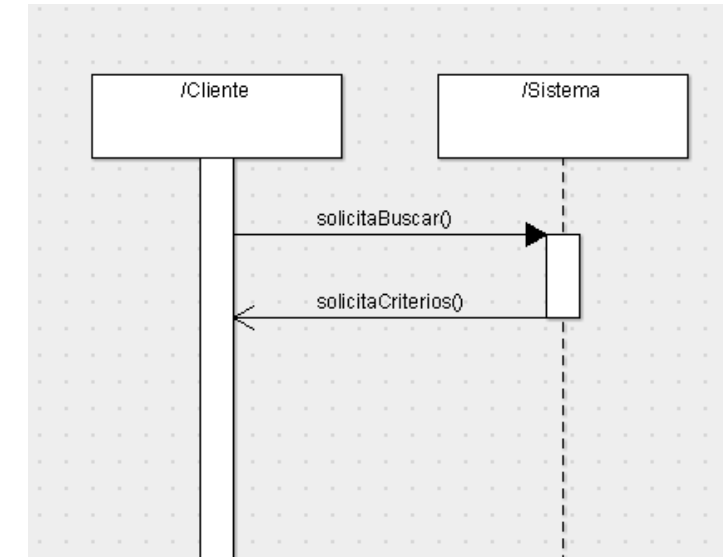
El segundo mensaje solicita al cliente los criterios de búsqueda en respuesta al evento anterior.



# Diagramas de secuencia.

## ❑ Tipos de mensajes:

- ❑ **Síncrono:** cuando se envía un mensaje a un objeto, no se recibe el control hasta que el objeto receptor ha finalizado la ejecución.
- ❑ **Asíncrono:** quien envía el mensaje continúa con su trabajo después de enviado, es decir, no necesita esperar la finalización del mismo en el objeto receptor. Este tipo de mensaje se representa mediante una flecha con la punta acierta o con media punta de flecha abierta
- ❑ **De retorno:** mensaje de confirmación.



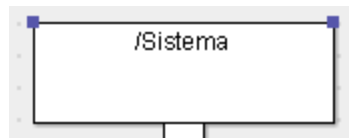
# Diagramas de secuencia.

Existe una nomenclatura para dar nombre a las entidades que representan los rectángulos, es decir, los objetos del diagrama:

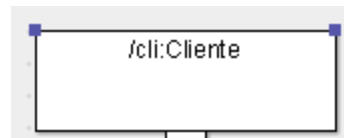
**nombre : tipo**

Tanto nombre como tipo pueden omitirse, pero no ambos a la vez. El nombre representa el objeto, el tipo el tipo del objeto (la clase)

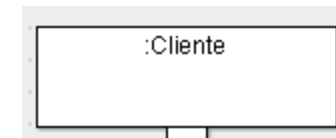
Ejemplos:



Una clase



El objeto cli de la clase Cliente



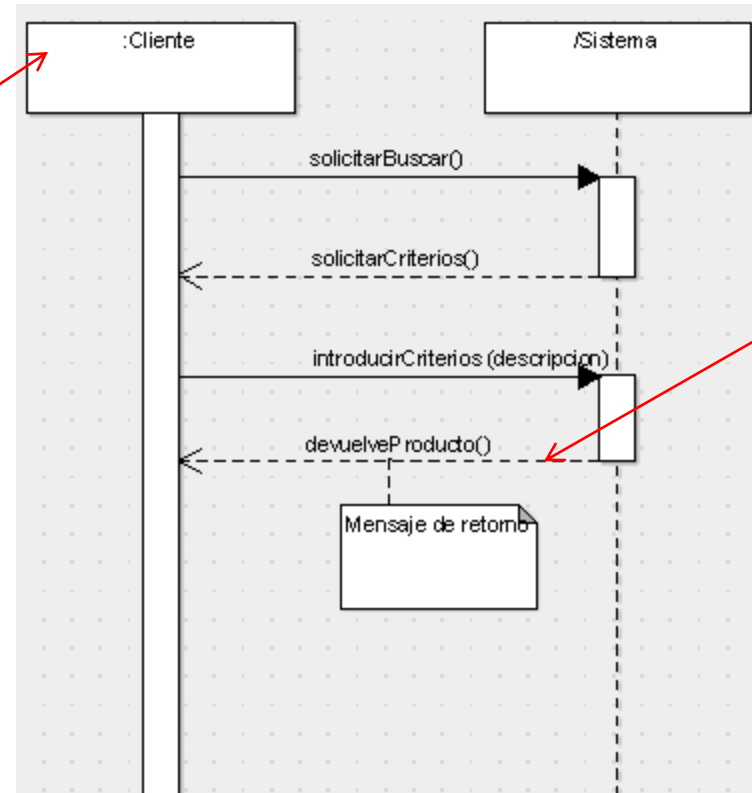
Un objeto cualquiera de la clase Cliente

# Diagramas de secuencia.

## ❑ Ejemplo: Diagrama de secuencia del caso de uso BuscarProductos

:Cliente

Significa un objeto  
cualquier de la  
clase Cliente



Mensaje de retorno

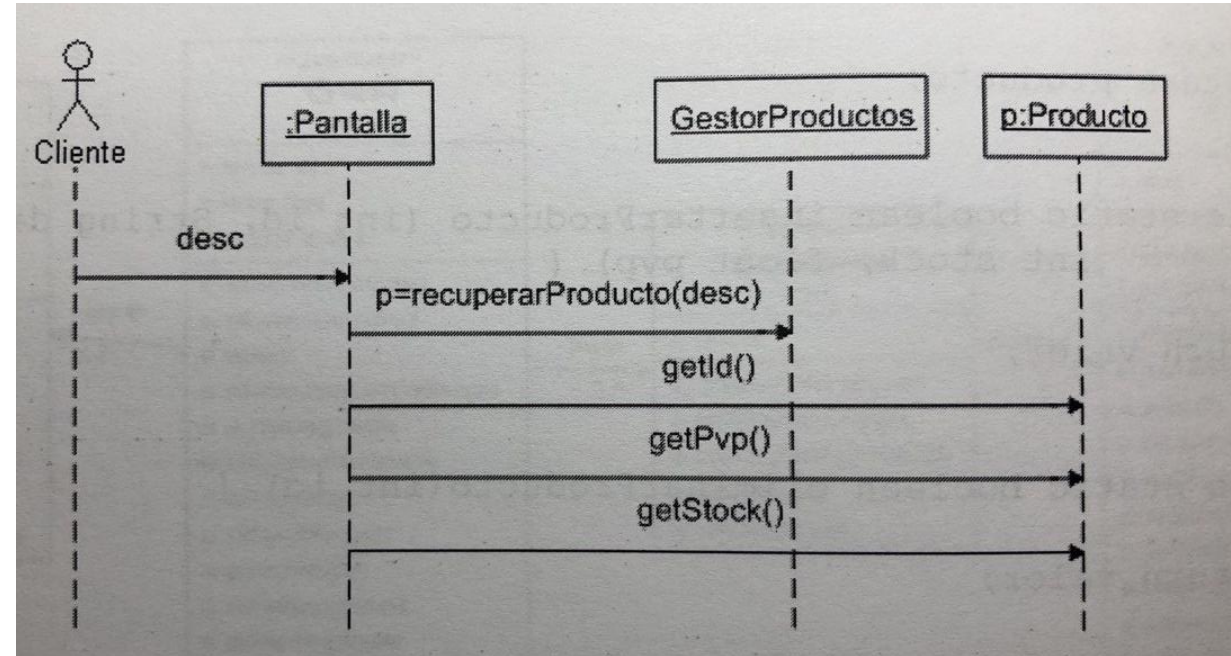
# Diagramas de secuencia.

## ❑ Ejemplo:

Supongamos un sistema de venta con varias clases: Cliente, Ventas, Producto, LineaDeVenta. Un cliente puede tener muchas ventas y una venta muchas líneas de venta y cada línea de venta tiene un producto.

También se ha definido la clase Pantalla que contiene la interfaz gráfica que permite la comunicación del cliente con el sistema.

El diagrama de secuencia del caso de uso BuscarProductos podría ser el representado en la imagen.

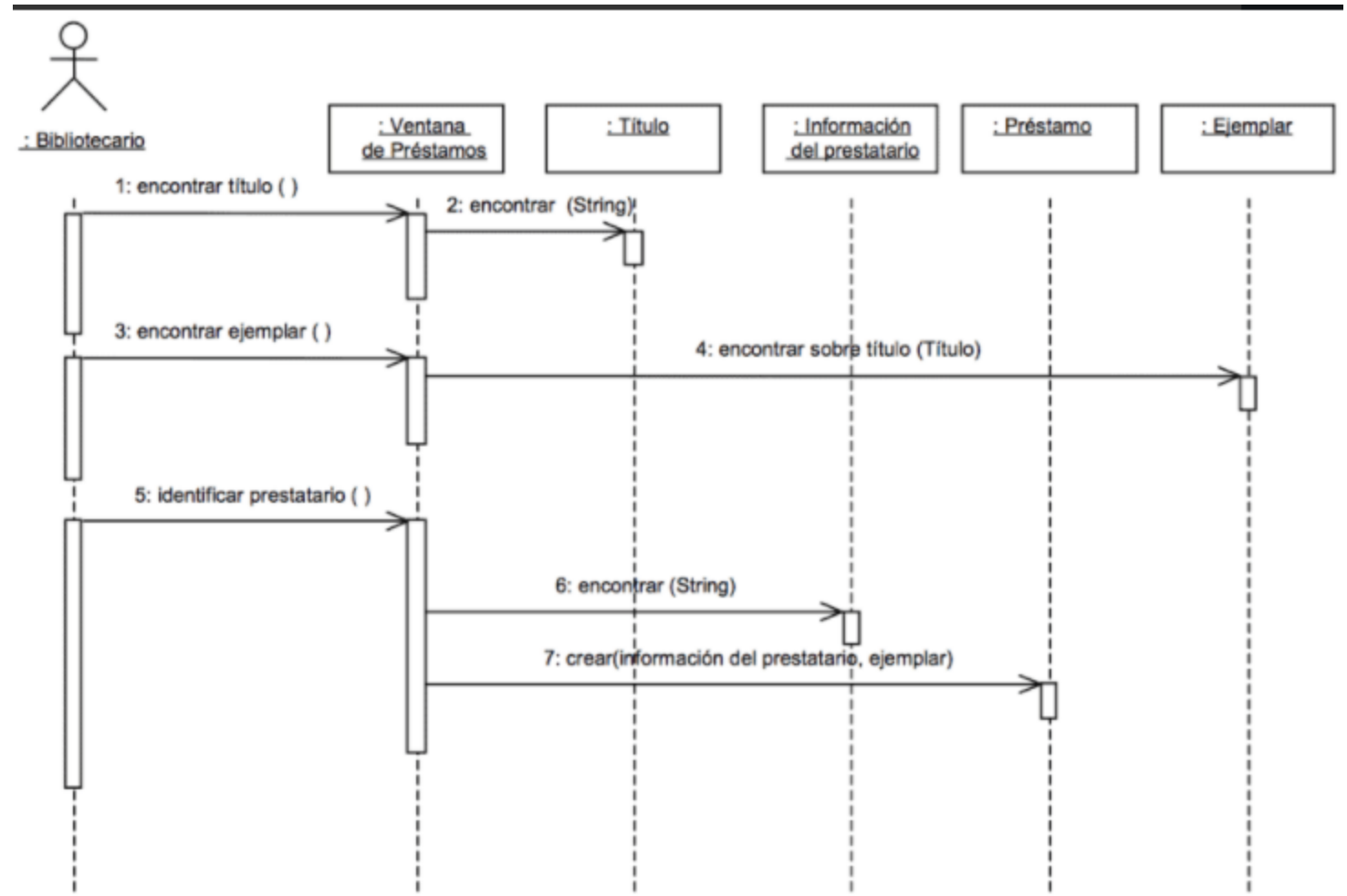


El cliente escribe la descripción del producto en la pantalla  
El objeto pantalla envía el mensaje recuperarProducto(desc) a la clase GestorProductos pidiendo los datos. Esta operación crea y devuelve un objeto p Producto. El objeto pantalla envia mensajes getId(), getPvp(), getStock() al objeto producto.

# Diagramas de secuencia.

## ❑ Ejemplo:

**Diagrama de secuencia** para el caso de uso: Prestar un ejemplar de una aplicación encargada de los préstamos y reservas de una biblioteca:





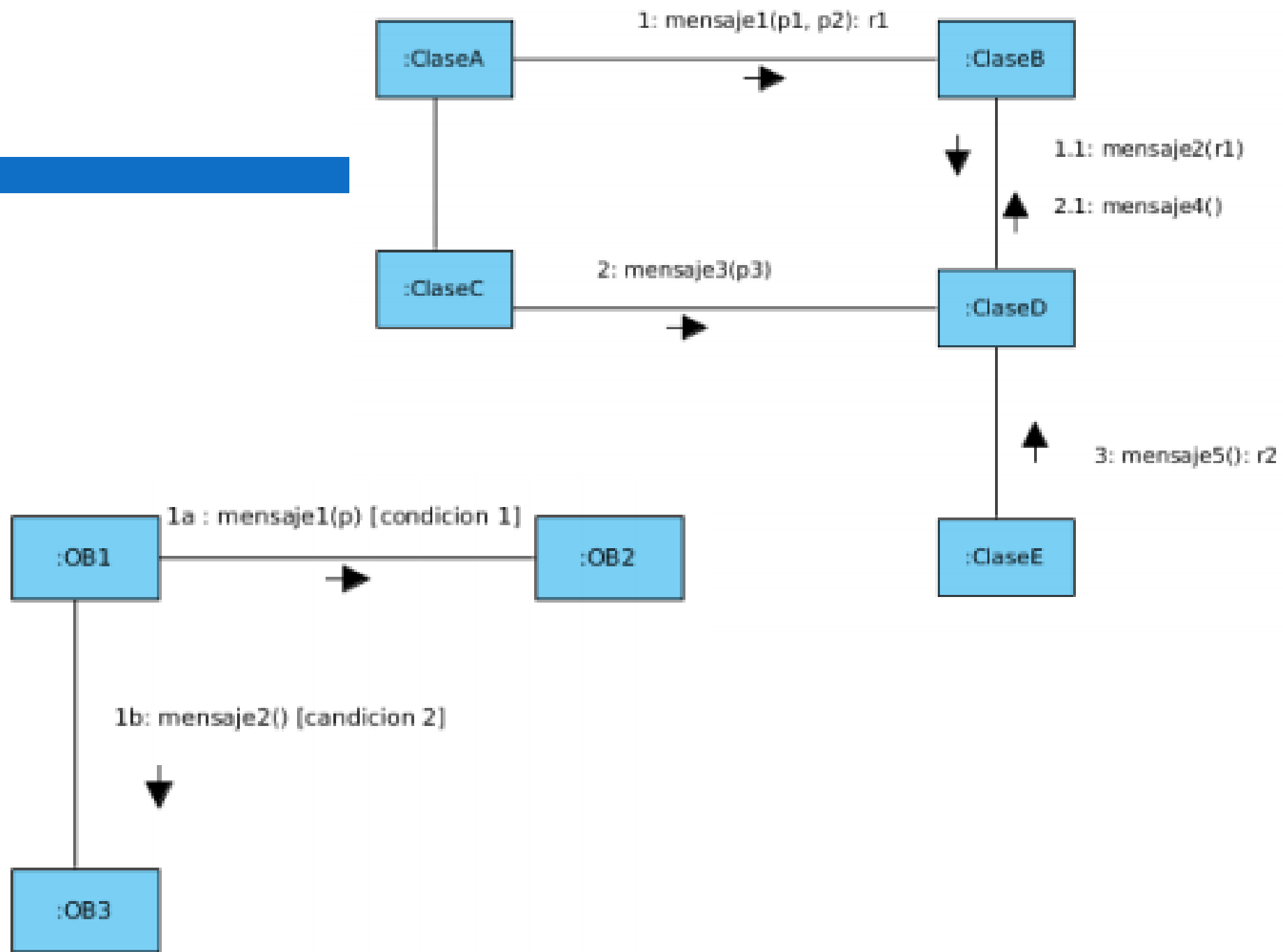
## **3.2 Diagramas de Colaboración (o Comunicación en UML 2)**

# Diagramas de colaboración.

- ❑ Muestran **cómo interactúan los objetos del sistema, a través del paso de mensajes.**
- ❑ Representa la misma información que los diagramas de secuencia pero de forma diferente.
- ❑ A diferencia de los diagramas de secuencia **se centra en la organización estructural de los objetos que envían y reciben mensajes.**
- ❑ Representa la información en forma de **grafo**:
  - ❑ Los **nodos** del grafo son los **objetos** (clases, la interfaz, el sistema ...) y se representan mediante rectángulos en cuyo interior aparece el nombre de la clase o el nombre del objeto: nombre de la clase La colocación de los objetos es más flexible.
  - ❑ Las **aristas** representan los **mensajes** que intercambian los objetos.
  - ❑ Los **mensajes** se representan con una flecha junto al enlace al que está asociado. La dirección de la flecha indica el receptor del mensaje. Se numeran para determinar su orden en el tiempo.  
Los mensajes llevan asociado el número de secuencia del mensaje, existiendo diferentes notaciones para la numeración de los mismos.

Sintaxis de un mensaje

[nºsecuencia][\*] [Condición de guarda]{valorDevuelto} : mensaje (argumentos)



# Diagramas de colaboración.

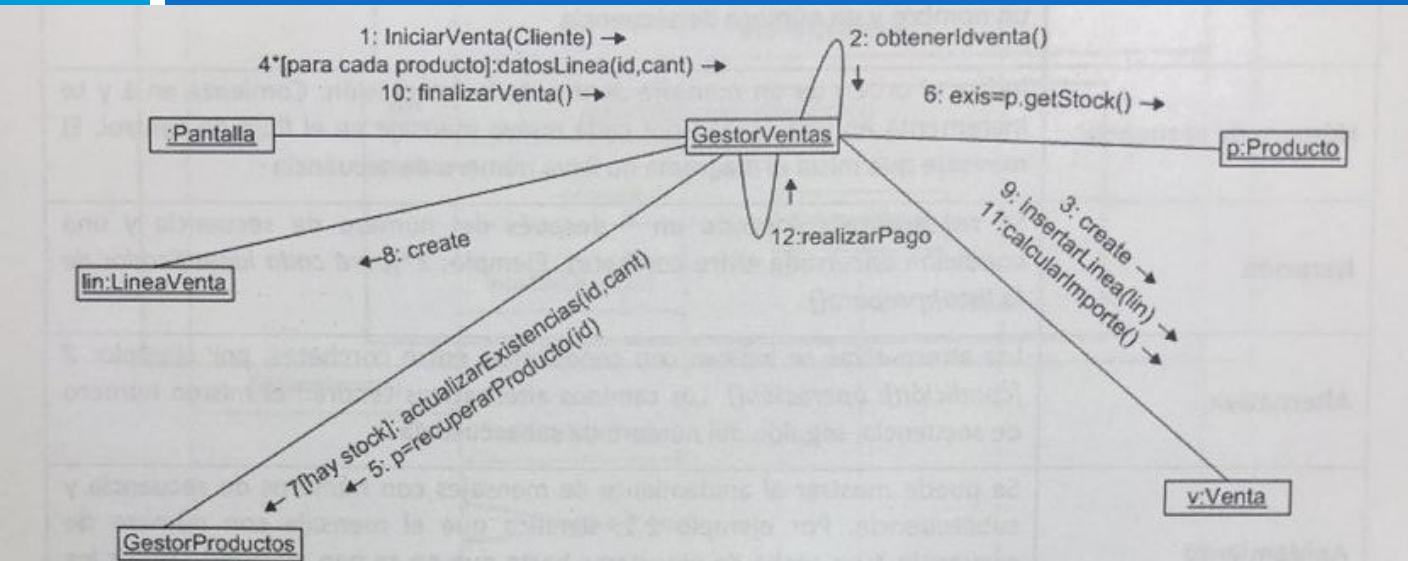


Diagrama de colaboración del caso de uso **ComprarProductos** con numeración simple

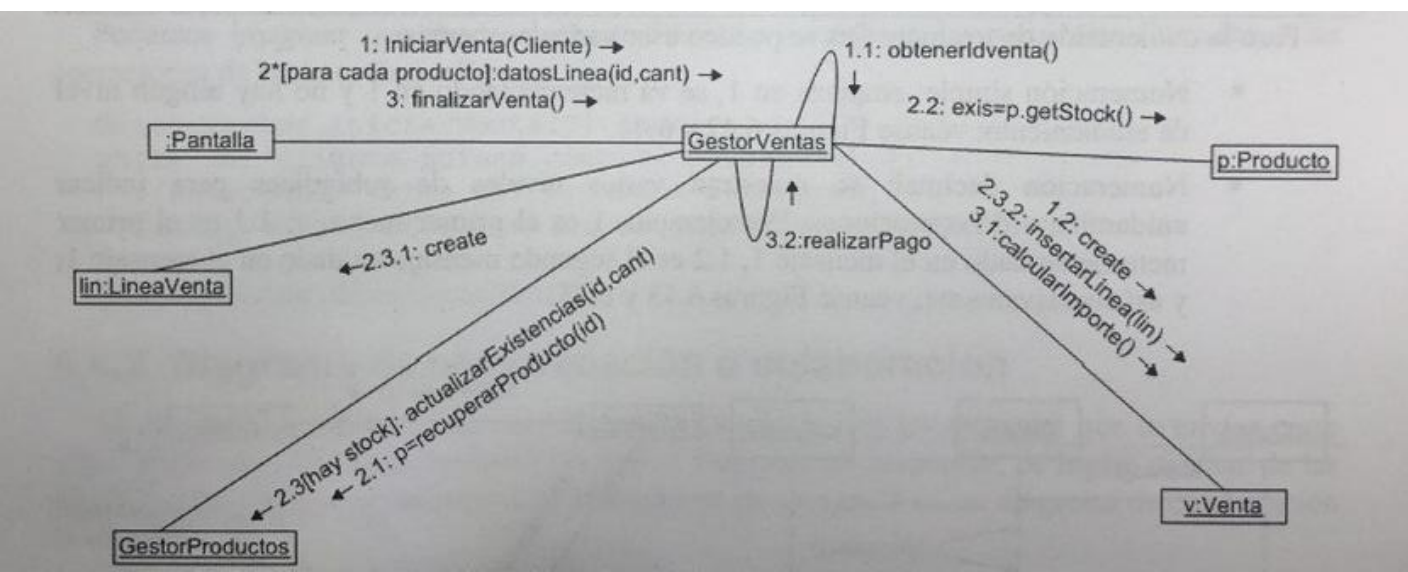


Diagrama de colaboración del caso de uso **ComprarProductos** con numeración decimal

Las iteraciones se representan con \*  
Las alternativas entre [ ]

# Diagramas de colaboración.

## ❑ Ejemplo:

**Diagrama de colaboración**  
para el caso de uso: **Prestar un  
ejemplar** de una aplicación  
encargada de los préstamos y  
reservas de una biblioteca.

