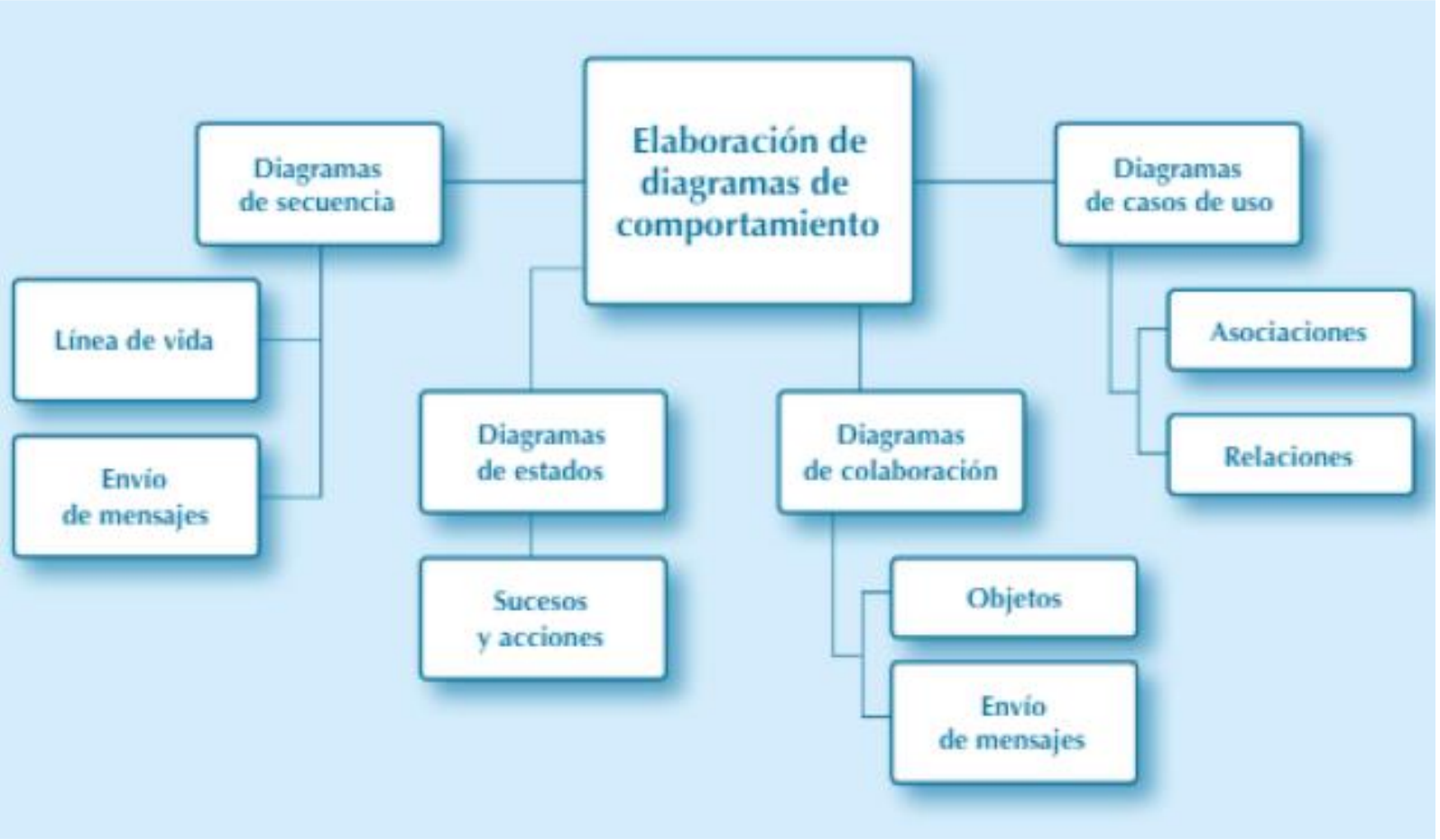
UNIDAD 6-2. ELABORACIÓN DE DIAGRAMAS DE COMPORTAMIENTO

VICENT MARTÍ

OBJETIVOS

- Los diagramas de clases ayudan a ver el problema con otra perspectiva y descubrir información nueva, pero siempre bajo una visión estática del sistema.
- En esta unidad, tendremos una visión dinámica, 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.
- Éste tipo de información la manejamos con los diagramas de comportamiento que incluyen: diagramas de casos de uso, diagramas de actividad, diagramas de estado, etc.
- Todos estos diagramas requieren seguir los pasos en espiral, pues surgen modificaciones en el proceso que debemos incorporar en refinamientos sucesivos, hasta la entrega del diseño.

MAPA CONCEPTUAL



CONTENIDO

1. Diagramas de comportamiento
2. Diagramas de casos de uso
3. Diagramas de secuencia
4. Diagramas de colaboración
5. Diagramas de estados
6. Diagramas de actividad

1. DIAGRAMAS DE COMPORTAMIENTO

Un diagrama de clases para un problema determinado 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 secuencia.
- Diagramas de colaboración.
- Diagramas de actividad.
- Diagramas de máquinas de estado.
- Otros.

2. DIAGRAMAS DE CASOS DE USO I

- Son un elemento fundamental del 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.

- Se documenta 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** (*acciones fundamentales que debe realizar el software al recibir información, procesarla y producir resultados. Suelen venir definidos por el cliente*), es decir, representan las funciones que un sistema puede ejecutar.

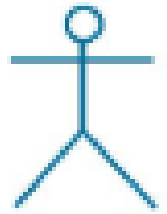
- Es una visualización gráfica de los requisitos funcionales del sistema, que está formado por casos de uso (se representan como elipses) y los actores que interactúan con ellos (se representan como monigotes). Su principal **función** es dirigir el proceso de creación del software, definiendo qué se 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.

2. DIAGRAMAS DE CASOS DE USO II

Los diagramas de casos de uso se crean en las primera etapa de desarrollo del software , y se enmarcan en el proceso de análisis (*normalmente se realiza antes que el diagrama de clases*) , para definir de forma detallada la funcionalidad que se espera cumpla el software, y que, además, se pueda comunicar fácilmente al usuario, pero, además se desprenden otras funciones que describen tanto la estructura del sistema como su comportamiento, lo que influye directamente en la implementación (*paso a código de las especificaciones software*) 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.

PRINCIPALES ELEMENTOS DE CASOS DE USO I

- 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. Siempre hay que intentar independizar los actores de la forma en que se interactúa con el sistema. Por ejemplo, un mismo usuario puede interpretar diferentes roles según cada operación que ejecute, cada rol representa un actor diferente. Suele ser interesante tener una lista de los usuarios reales para cada actor. Ejemplo: cajero, administrador, etc.



- 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***: no interactúan con el sistema pero desencadenan el trabajo de otro actor.

PRINCIPALES ELEMENTOS DE CASOS DE USO II

- El sistema: es todo lo que engloba el rectángulo. Es lo que se codificará en un futuro. Todo lo que queda fuera es ajeno al sistema, pero pueden interactuar con él, como los actores. En la parte superior a la derecha o izquierda suele tener el nombre para identificarlo. Ejemplo: gestión de nóminas.
- Caso de uso: es lo que hace el sistema, la funcionalidad. Detrás de un caso de uso, habrá uno o varios métodos de diferentes clases. Se utiliza una elipse que en su interior tiene el nombre de la funcionalidad a ejecutar. Ejemplo borrar usuario, enviar informe, etc.

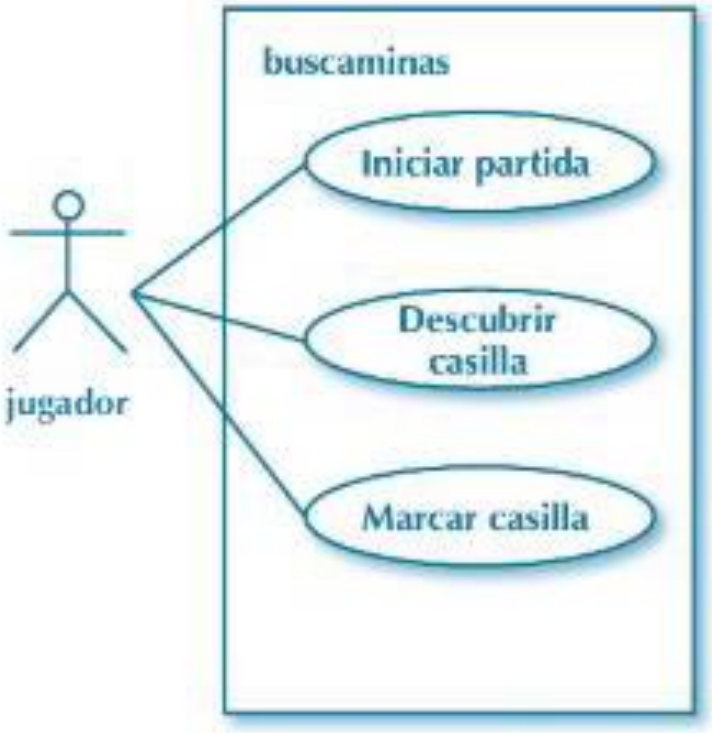
PRINCIPALES ELEMENTOS DE CASOS DE USO III

- Por ejemplo: en un restaurante de comida rápida, un cajero puede ser también supervisor o administrador. Vemos los casos de uso (comer, pagar, etc.) que tiene cada actor dentro del sistema (restaurante). Si hay una persona que realiza distintos roles, no debemos agrupar los roles en un único actor, puesto que, en un futuro, dichas tareas podrían ser realizadas por distintas personas o actores.

PRINCIPALES ELEMENTOS DE CASOS DE USO IV

- Buscaminas: Los casos de uso que se identifican a simple vista en dicho juego serían los siguientes.

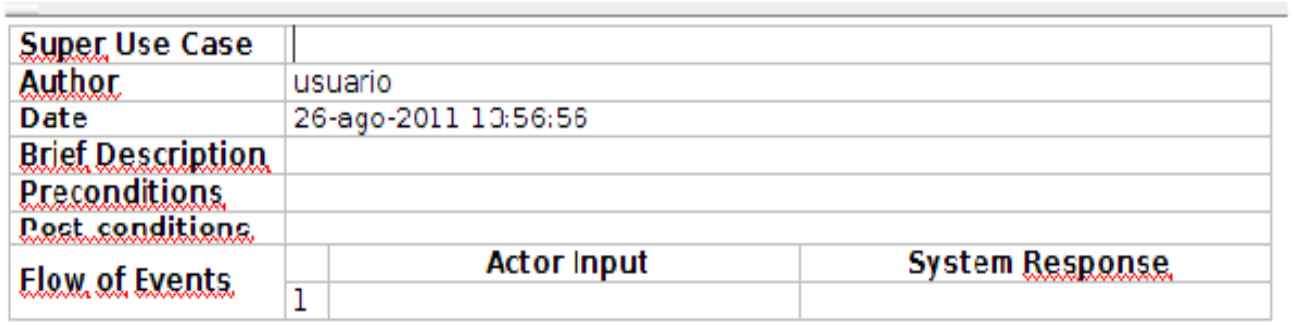
a) Iniciar partida
b) Descubrir casilla
c) Marcar casilla



PRINCIPALES ELEMENTOS DE CASOS DE USO V

El objetivo principal es 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. Tendremos una descripción textual, en la que se deben incluir, al menos, los siguientes datos (a los que se denomina contrato, vemos el ejemplo obtenido de la herramienta Visual Paradigm):

- **Nombre:** nombre del caso de uso.
- **Actores:** interactúan con el sistema a través del caso de uso.
- **Propósito:** breve descripción de lo que se espera que haga.
- **Precondiciones:** deben cumplirse para llevar a cabo el caso de uso.



Super Use Case			
Author	usuario		
Date	26-ago-2011 10:56:56		
Brief Description			
Preconditions			
Post conditions			
Flow of Events		Actor Input	System Response
	1		

- **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.
- **Postcondiciones:** las que se cumplen una vez que se ha realizado el caso de uso.

RELACIONES

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. 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:

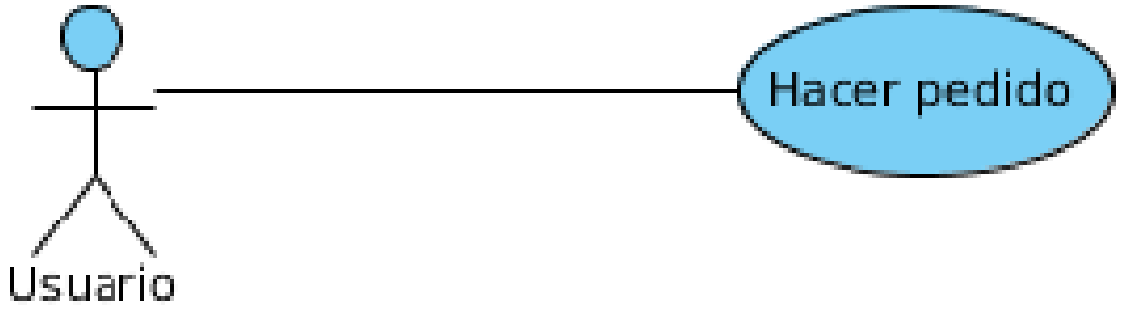
- **Interacción o asociación**: representa la relación entre el actor que lo inicia y el caso de uso.
- **Generalización**: se utiliza para representar relaciones de herencia entre casos de uso o actores.
- **Extensión**: se utiliza para representar relaciones entre un caso de uso que requiere la ejecución de otro en determinadas circunstancias.
- **Inclusión**: se utiliza cuando queremos dividir una tarea de mayor envergadura en otras más sencillas, que son utilizadas por la primera. Representa una relación de uso, y son muy útiles cuando es necesario reutilizar tareas.

A continuación las vemos con un poco más de detalle.

INTERACCIÓN O ASOCIACIÓN

Representa la relación entre el actor que lo inicia y el caso de uso.

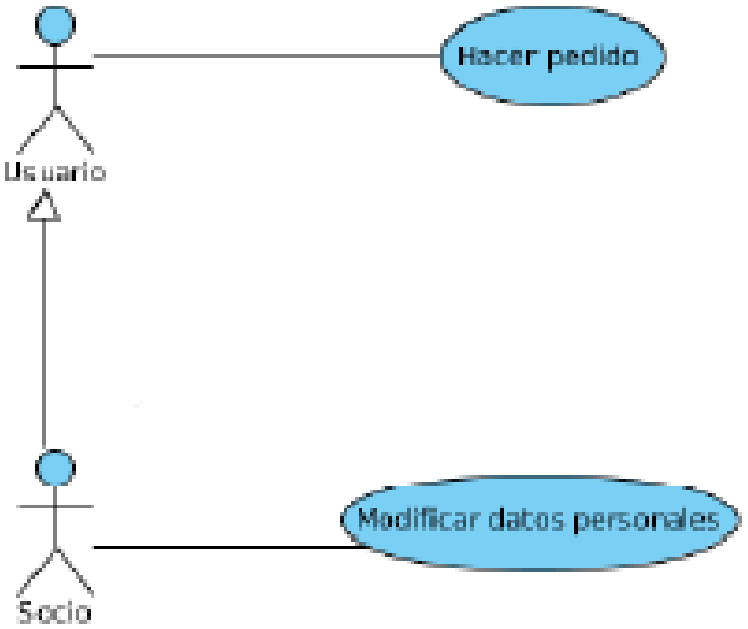
Hay una asociación entre un actor y un caso de uso si el actor interactúa con el sistema para llevar a cabo el caso de uso o para iniciarlo.
Una asociación se representa mediante un linea continua que une un actor con un caso de uso. Por ejemplo, un usuario de un sistema de venta por Internet puede hacer un pedido, lo que se representa del siguiente modo:



GENERALIZACIÓN

Se utiliza para representar relaciones de herencia entre casos de uso o actores.

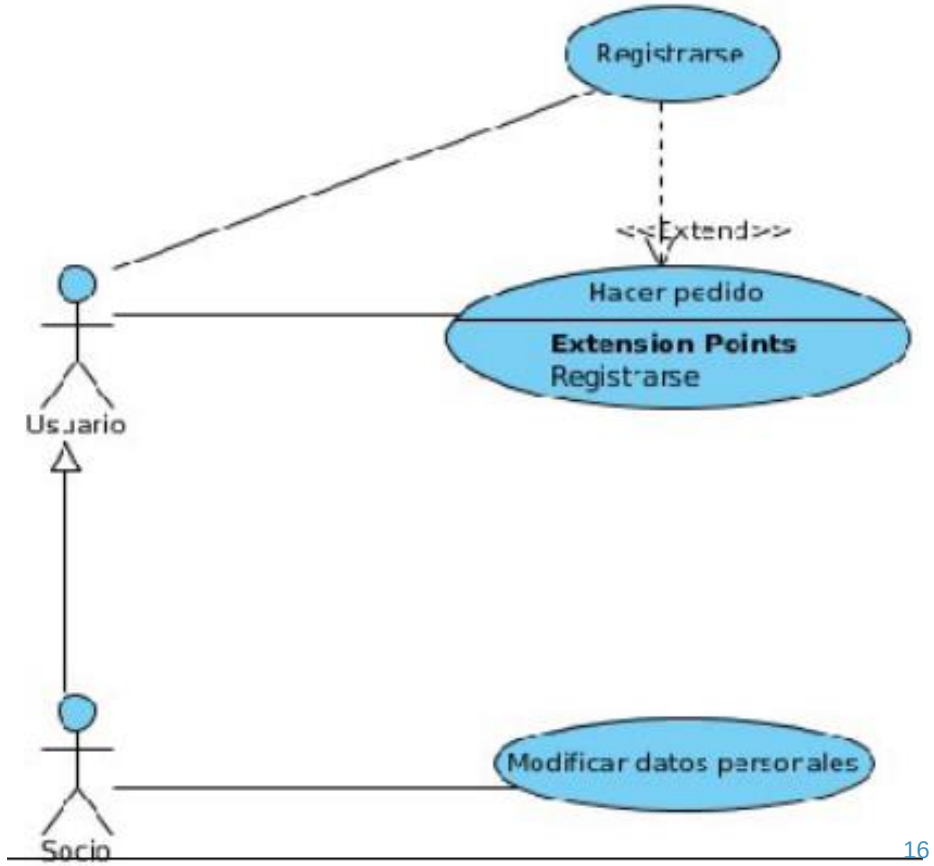
Es posible que, igual que con los diagramas de clases, existan casos de uso que tengan comportamientos semejantes a otros que los modifican o completan de alguna manera. El caso base se define de forma abstracta y los hijos heredan sus características añadiendo sus propios pasos o modificando alguno.
Por ejemplo, el usuario del sistema de venta por Internet puede a su vez darse de alta en la página web para que tengan sus datos registrados a la hora de hacer el pedido, en este caso el usuario es la generalización del socio. Ambos actores pueden hacer un pedido, pero solo el socio puede modificar sus datos en el sistema.



EXTENSIÓN

Se utiliza para representar relaciones entre un caso de uso que requiere la ejecución de otro en determinadas circunstancias. Tipo “extends” .

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. Por ejemplo, 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.



INCLUSIÓN I

Se utiliza cuando queremos dividir una tarea de mayor envergadura en otras más sencillas, que son utilizadas por la primera. Son muy útiles cuando es necesario reutilizar tareas. Tipo “include”.

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.
Las ventajas de esta asociación son:

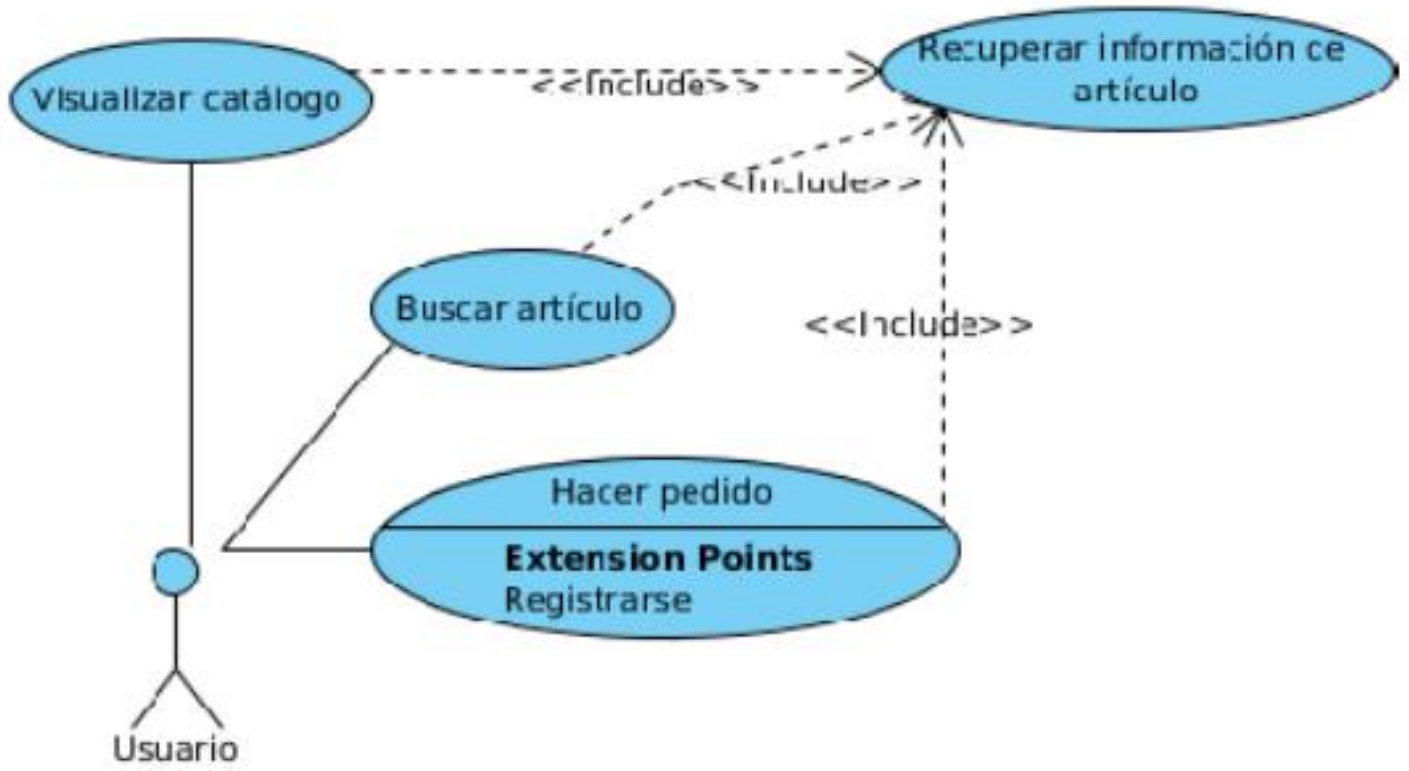
- Las descripciones de los casos de uso son más cortas y se entienden mejor.
- La identificación de funcionalidad común puede ayudar a descubrir el posible uso de componentes ya existentes en la implementación.

Las desventajas son:

- La inclusión de estas relaciones hace que los diagramas sean más difíciles de leer, sobre todo para los clientes.

INCLUSIÓN II

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.



TENER EN CUENTA

- Cuando usamos relaciones de **inclusión o extensión** no podemos olvidar que los casos de uso extendidos o incluidos deben representar un **flujo de actividad completo** desde el punto de vista de lo que un actor espera que el sistema haga por él.

- No utilizar estas herramientas sólo para descomponer un caso de uso de envergadura en otros más pequeños, **piedra angular del diseño estructurado y no del orientado a objetos*.***

- Si un caso de uso describe una variación de una conducta normal, utilizamos la extensión.

- Si necesitas evitar las repeticiones de un caso de uso, utiliza inclusión.

- No debemos incluir aspectos de diseño, mostrar lo que hace el sistema evitando indicar como será diseñado o programado.

EJEMPLO: SERVIR PEDIDO

Suponer 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?

- Interacción o asociación
- Generalización
- Extensión
- Inclusión

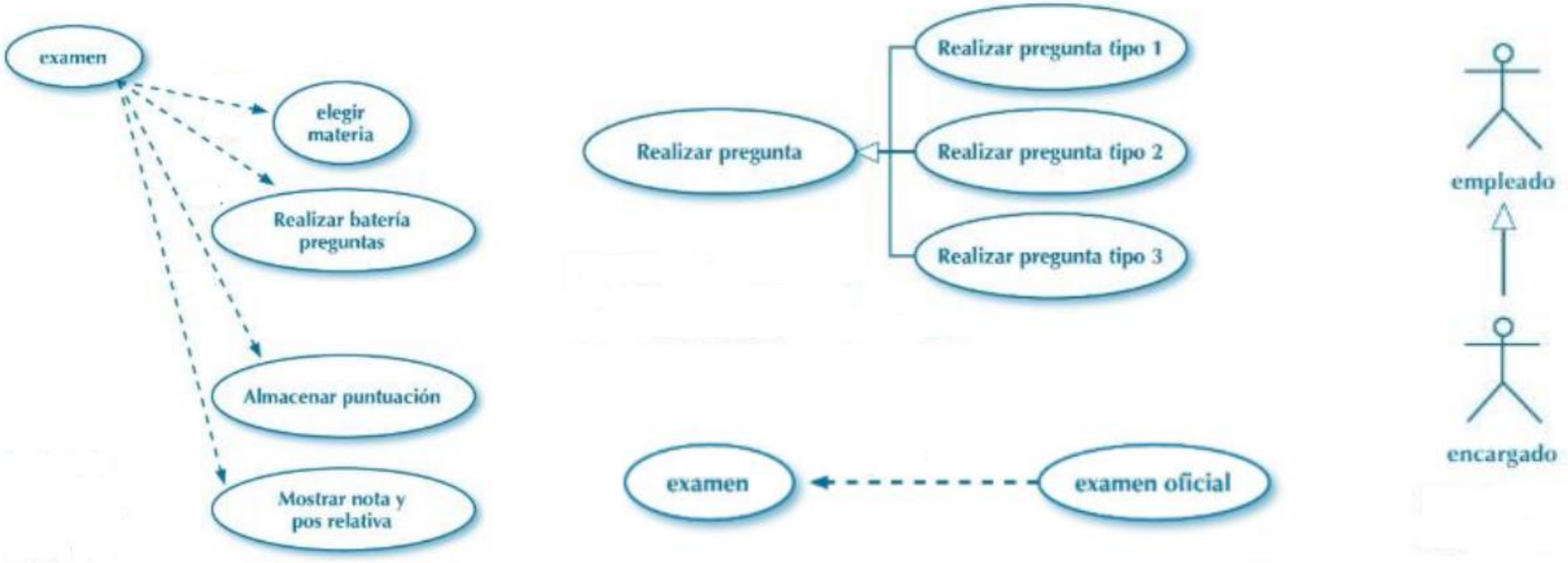
EJEMPLO: SERVIR PEDIDO. SOLUCIÓN

Suponer 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?

- Interacción o asociación
- Generalización
- Extensión
- **Inclusión**

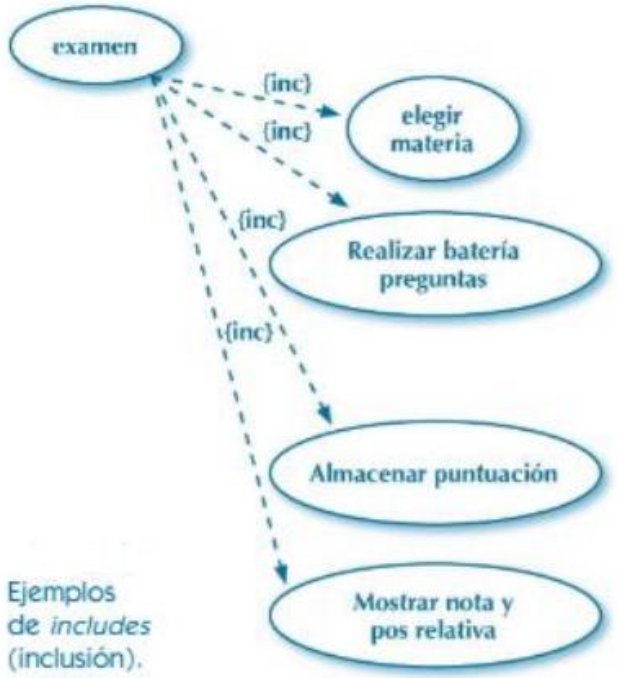
EJEMPLO: TIPO DE RELACIÓN

¿Qué tipo de relación emplearías en cada caso de uso?



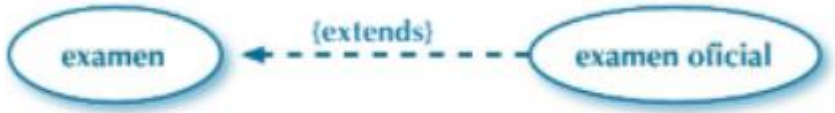
EJEMPLO: TIPO DE RELACIÓN. SOLUCIÓN

¿Qué tipo de relación emplearías en cada caso de uso?



Ejemplos de *includes* (inclusión).

Ejemplo en casos de uso de generalización o especialización.



Ejemplo de *extends* (extensión).

Ejemplo en actores de generalización o especialización.

3. DIAGRAMAS DE SECUENCIA

- Los diagramas de secuencia se utilizan para formalizar la descripción de un escenario o conjunto de ellos representando que mensajes fluyen en el sistema así como quien los envía y quien los recibe.

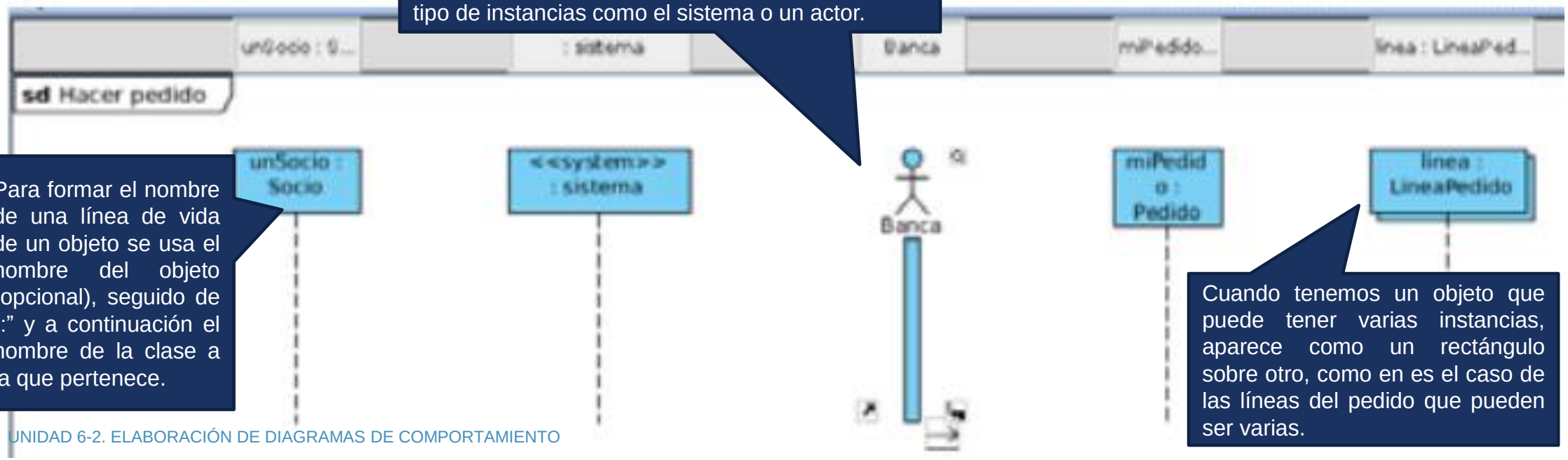
- Los objetos y actores que forman parte del escenario se representan mediante rectángulos distribuidos horizontalmente en la zona superior del diagrama, a los que se asocia una línea temporal vertical (una por cada actor) de las que salen, en orden, los diferentes mensajes que se pasan entre ellos. 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.

- 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 interaccionan a lo largo del tiempo.

REPRESENTACIÓN DE OBJETOS, LÍNEA DE VIDA Y PASO DE MENSAJES I

Representación de objetos y línea de vida.

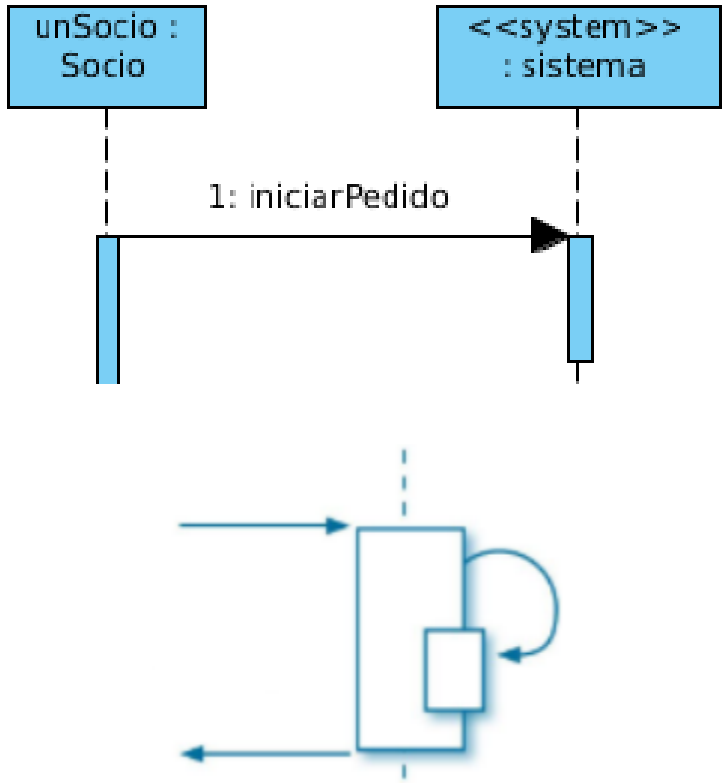
En un diagrama de secuencia se dibujan las entidades que participan dentro de rectángulos que se distribuyen horizontalmente. De cada entidad sale una línea de puntos que representa el paso del tiempo, se les denomina línea de vida y representan que el objeto existe. Usaremos el sistema para representar solicitudes al mismo, como por ejemplo pulsar un botón para abrir una ventana o una llamada a una subrutina.



REPRESENTACIÓN DE OBJETOS, LÍNEA DE VIDA Y PASO DE MENSAJES II

Mensajes: Invocación de métodos.

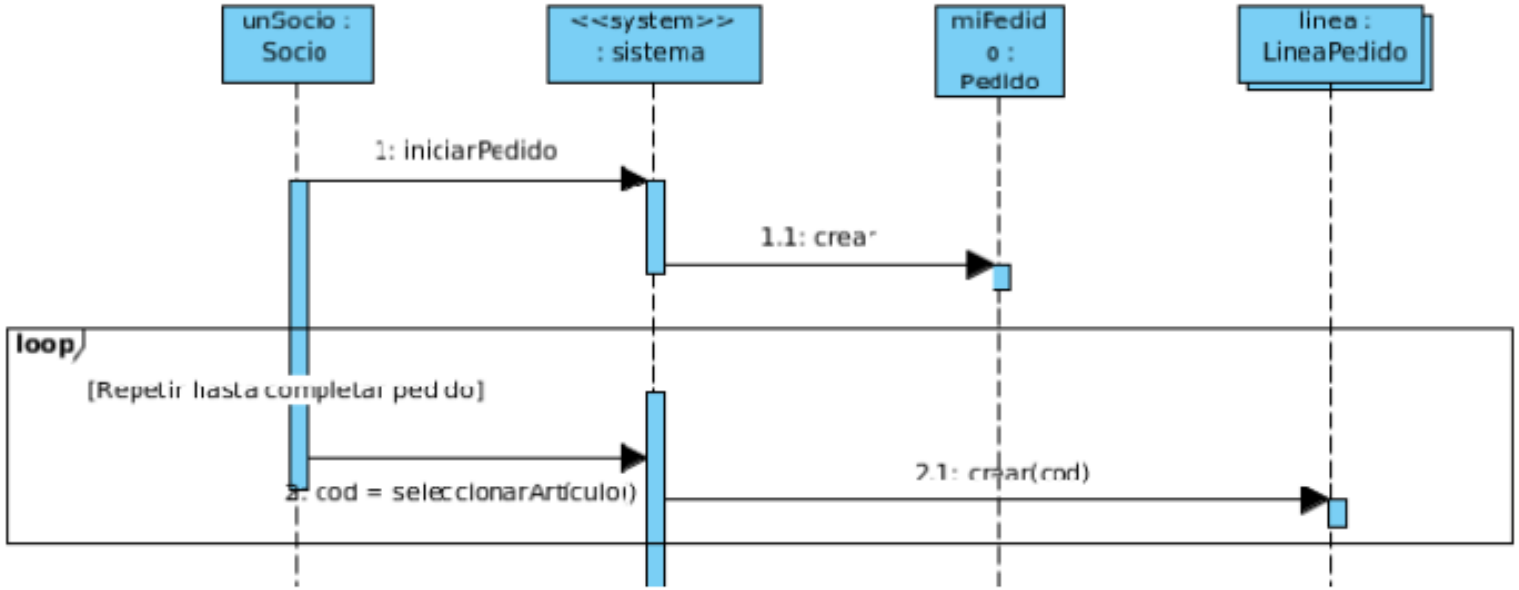
Los mensajes, que significan la invocación de métodos, se representan como flechas horizontales que van de una línea de vida a otra, indicando con la flecha la dirección del mensaje, los extremos de cada mensaje se conectan con una línea de vida que conecta a las entidades al principio del diagrama. Los mensajes se dibujan desde el objeto que envía el mensaje al que lo recibe, pudiendo ser ambos el mismo objeto y su orden 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. Un caso especial es la **recursividad** que podemos expresar con una flecha que sale de un cuadro más grande y se dirige a uno más pequeño superpuesto.



REPRESENTACIÓN DE OBJETOS, LÍNEA DE VIDA Y PASO DE MENSAJES III

Mensajes: Iteraciones y condicionales.

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. 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.



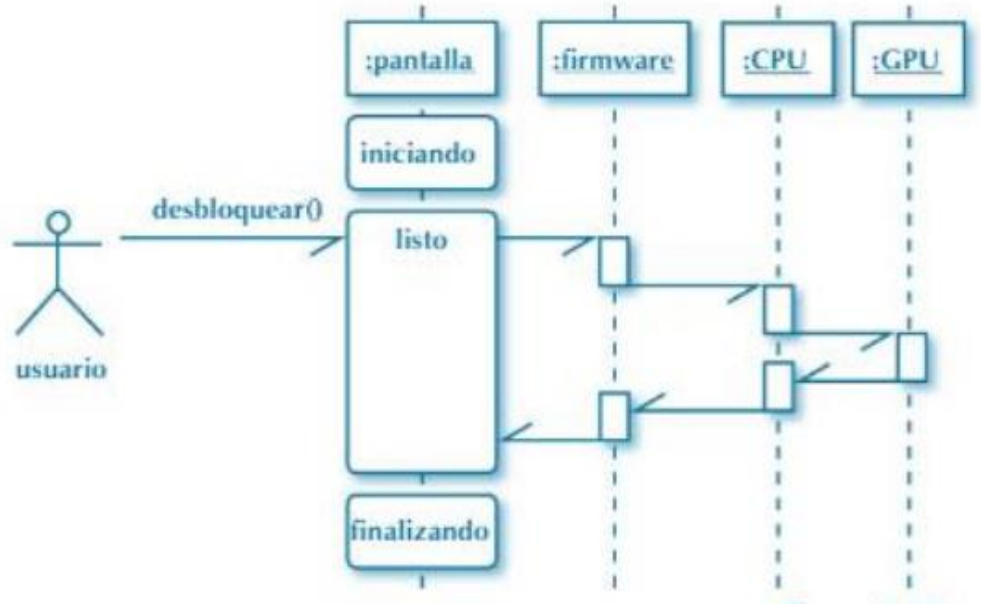
TIPOS DE MENSAJES

- **Simples:** Pasan el hilo de ejecución de un objeto a otro. Esto quiere decir que, tras el envío del mensaje, el código del objeto no volverá a ejecutarse, a menos que exista la recepción de otro mensaje.
- **Sincrónicos**: Esperan a que el método del otro objeto invocado termine su tarea para poder seguir la ejecución del método original.
- **Asincrónicos:** No impiden que el método origen siga su ejecución normal. Una vez termina el mensaje invocado, el objeto llamador recibirá una notificación de que la llamada ha terminado y ejecutará el código que esté programado al respecto.

EJEMPLO: DIAGRAMA DE SECUENCIA

Imaginemos el siguiente sistema: un dispositivo con una pantalla táctil que responde a las interacciones del usuario. Este dispositivo tiene un firmware instalado, el cual responderá a los eventos iniciados por el usuario realizando las tareas oportunas. El firmware hace uso del procesador o CPU del dispositivo para ejecutar las órdenes del usuario

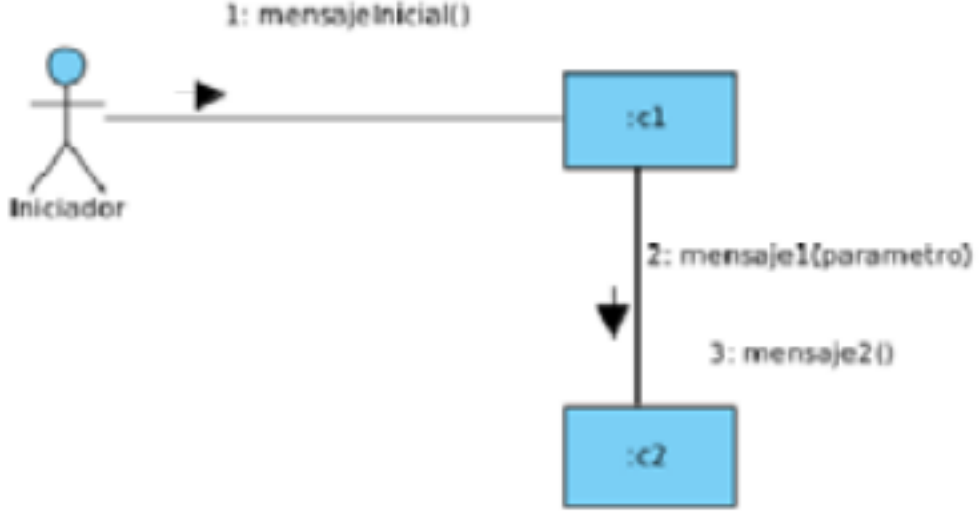
Por su parte, la CPU, cuando ejecutan cualquier tarea que requiera una modificación de la interfaz gráfica (prácticamente la mayoría de las acciones ejecutadas por el usuario), lo que hará es delegar en la GPU o procesador gráfico todas las tareas que impliquen un cambio de lo que el usuario está visualizando. Además podemos incluir los estados dentro del esquema de secuencias, de este modo proporciona más información. La secuencia empieza y termina en el objeto pantalla. Vemos como se complementa un diagrama de uso con el de secuencia.



4. DIAGRAMAS DE COLABORACIÓN I

- Los diagramas de colaboración son un complemento para los de secuencia cuyo objetivo es mostrar las interacciones entre los objetos del diagrama mediante el paso de mensajes entre ellos.
- 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.
- A diferencia de los diagramas de secuencia, no se incluye una línea temporal, los mensajes son numerados para determinar su orden el tiempo.

En este diagrama de colaboración el actor Iniciador manda un mensaje al objeto c1 que inicia el escenario, a continuación el objeto c1 envía el mensaje mensaje1 que lleva un parámetro al objeto c2 y después el mensaje mensaje2, que no tiene parámetros de nuevo al objeto c2.



4. DIAGRAMAS DE COLABORACIÓN II

Los diagramas de colaboración y secuencia utilizan los mismos elementos pero distribuyéndolos de forma diferente, en 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.

REPRESENTACIÓN DE OBJETOS

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, algunos lo subrayan.

: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.

PASO DE MENSAJES

Para que sea posible el paso de mensajes es necesario que exista una asociación entre los objetos. En la imagen es posible el paso de mensajes entre el objeto objeto1 y objeto2, además de quedar garantizada la navegación y visibilidad entre ambos. Un mensaje es la especificación de una comunicación entre objetos que transmite información y desencadena una acción en el objeto destinatario. La sintaxis de un mensaje es la siguiente:

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

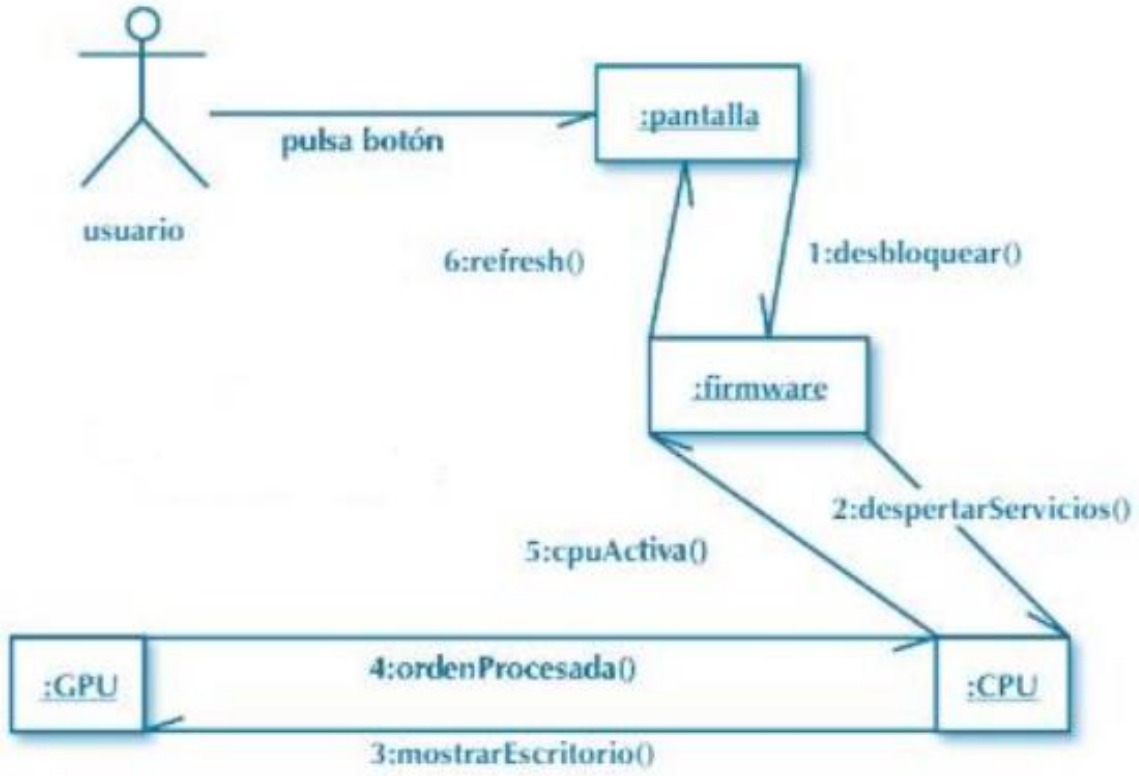
Secuencia: representa el nivel de anidamiento del envío del mensaje dentro de la interacción. Los mensajes se numeran para indicar el orden en el que se envían, y si es necesario se puede indicar anidamiento incluyendo subrangos.
*: indica que el mensaje es iterativo.
Condición de guarda: debe cumplirse 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.

EJEMPLO: DIAGRAMA DE COLABORACIÓN

En el siguiente sistema: un dispositivo con una pantalla táctil que responde a las interacciones del usuario. Este dispositivo tiene un firmware instalado, el cual responderá a los eventos iniciados por el usuario realizando las tareas oportunas, en este caso pulsa un botón de desbloquear.

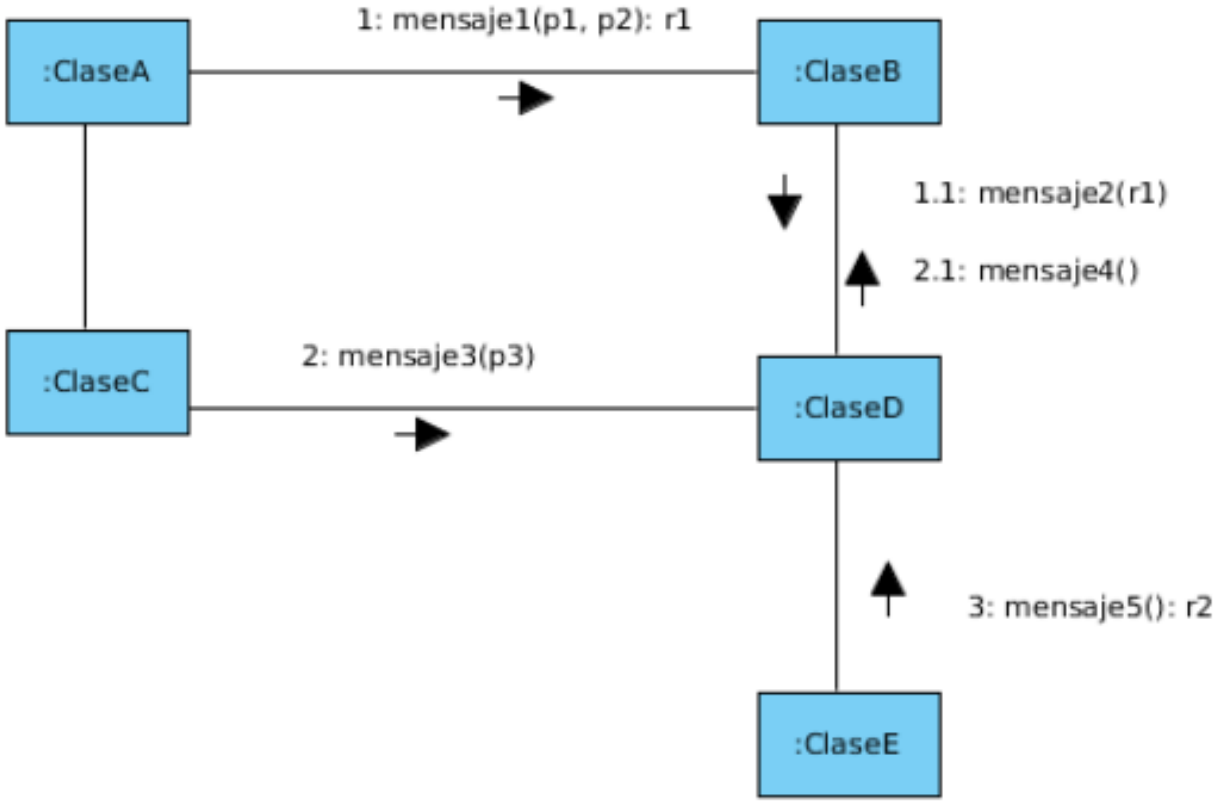
El firmware hace uso del procesador o CPU del dispositivo para ejecutar las órdenes del usuario. Por su parte, la CPU, lo que hará es delegar en la GPU o procesador gráfico todas las tareas que impliquen un cambio de lo que el usuario está visualizando.

Los pasos de mensajes contienen un número de secuencia, dos puntos (:) y el nombre del método invocado. Podemos observar que la información que proporciona es similar a la de los diagramas de secuencia.



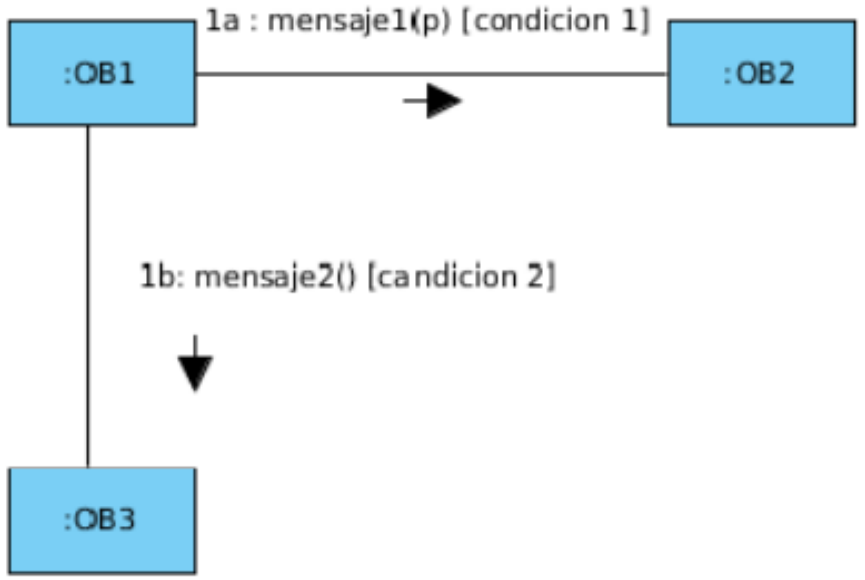
EJEMPLO: DIAGRAMA DE COLABORACIÓN

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, 2 y 3 tiene parámetros y los mensajes 1 y 5 devuelve un resultado.



EJEMPLO: DIAGRAMA DE COLABORACIÓN

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

5. DIAGRAMAS DE ESTADO I

En la POO, muchas veces, dependiendo de los sucesos o eventos que ocurran y del tiempo, los objetos o el sistema están o quedarán en un estado concreto. Existen muchos tipos de eventos como por ejemplo:

- Salta una alarma
- Alguien pulsa un botón
- Se detiene un servicio en un servidor
- Termina la ejecución de un proceso y se queda inactivo

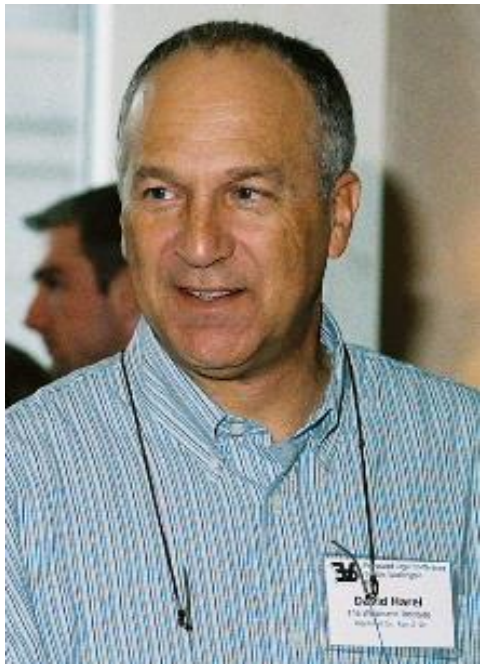
- El rectángulo representa el estado.
- El punto inicial: el inicio de una serie de estados
- El punto final: representa la finalización de una serie de estados.
- Las flechas representan las transiciones.

5. DIAGRAMAS DE ESTADO II

El diagrama de estados está basado en los statecharts de **David Harel**.

Representan máquinas de estados (**autómatas de estados finitos**). Modelo matemático que realiza cómputos sobre una entrada para producir una salida. Se representa mediante un grafo cerrado y dirigido cuyos vértices son estados y sus aristas son transiciones, que van etiquetadas con un símbolo que representa el evento que desencadena la transición. Parte de un estado inicial y termina en uno o varios estados finales.

Para modelar el comportamiento dinámico basado en la respuesta a determinados eventos de aquellos objetos que requieran su especificación, normalmente por su comportamiento significativo en tiempo real y su participación en varios casos de uso. El resto de objetos se dice que tienen un único estado.



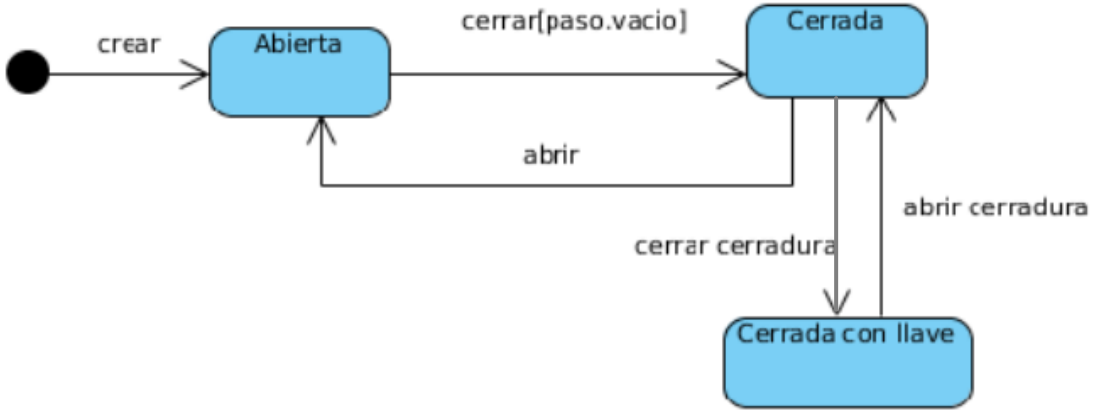
David Harel

5. DIAGRAMAS DE ESTADO III

En relación con el diagrama de estados se cumple que:

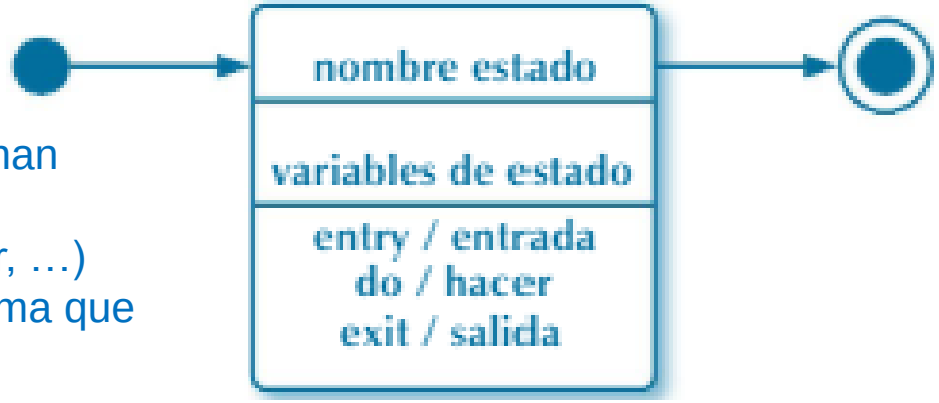
- Un objeto está en un estado concreto en un cierto momento, que viene determinado, parcialmente, por los valores de sus atributos.
- La transición de un estado a otro es momentánea y se produce cuando ocurre un evento.
- Una máquina de estados procesa un evento cada vez y termina con todas las consecuencias del evento antes de procesar otro. Si ocurren dos eventos simultáneamente se procesan como si se hubieran producido en cualquier orden, sin pérdida de generalidad.

Un diagrama de máquina de estados expresa el comportamiento de un objeto como una progresión a través de una serie de estados, provocada por eventos y las acciones relacionadas que pueden ocurrir. Por ejemplo, aquí tenemos el diagrama de estados de una puerta. **Analiza el diagrama de estados de la puerta. ¿Se puede abrir directamente, una puerta que está cerrada con llave?**



ESTADOS

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



Elementos de un estado:

- Nombre
- Variables de estado: similares a los atributos de una clase. Almacenan información.
- Actividad, son sucesos y acciones a realizar. (entrada, salida, hacer, …)
- Subestados, cuando el estado es complejo y necesita de un diagrama que lo defina.
- Eventos diferidos.

Existen dos tipos de estado especiales, estado inicial y estado final.

Estado inicial: es un pseudoestado que indica el punto de partida por defecto para una transición cuyo destino es el límite de un estado compuesto. El estado inicial del estado de nivel más alto representa la creación de una nueva instancia de la clase.

Estado final: Estado especial dentro de un estado compuesto que, cuando está activo, indica que la ejecución del estado compuesto ha terminado y que una transición de finalización que sale del estado compuesto está activada.

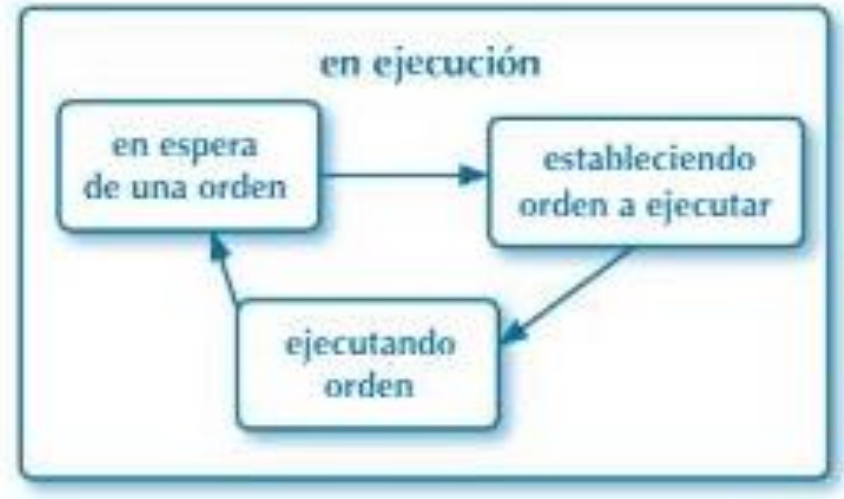
EJEMPLO DE DIAGRAMA DE ESTADOS

En ocasiones algunos dispositivos están en modo reposo o alerta esperando que ocurra un evento en el sistema para activarse en modo normal. Es muy común actualmente para reducir el consumo de batería. Estos estados son reversibles tras sacudir el móvil, pulsar una tecla o botón, tocar la pantalla, etc. Puede observarse en ella el estado standby se activará tras 5 segundos sin actividad con el dispositivo.

EJEMPLO DE SUBESTADOS

En ocasiones, un estado es una agrupación de estados, siguiendo con el ejemplo anterior:

- En espera de una orden del usuario. En este caso el móvil estaría esperando que el usuario realice una acción determinada como hacer scroll, pulsar en la pantalla, etc.
- Estableciendo la orden a ejecutar. Dependiendo del gesto del usuario, el firmware del móvil determinará que proceso tiene que lanzar y con qué parámetros.
- Ejecutando la acción. Por último , el sistema ejecutará la acción demandada por el usuario.



EVENTOS

Un evento es un acontecimiento que ocupa un lugar en el tiempo y espacio que funciona como un estímulo que dispara una transición en una máquina de estados. Existen eventos externos y eventos internos según el agente que los produzca. Tipos de eventos:

Señales (excepciones): la recepción de una señal, que es una entidad a la que se ha dado nombre explícitamente (clase estereotipada), prevista para la comunicación explícita - y asíncrona- entre objetos. Es enviada por un objeto a otro objeto o conjunto de objetos. Las señales con nombre que puede recibir un objeto se modelan designándolas en un compartimento extra de la clase de ese objeto. Normalmente una señal es manejada por la máquina de estados del objeto receptor y puede disparar una transición en la máquina de estados.
Llamadas: la recepción de una petición para invocar una operación. Normalmente un evento de llamada es modelado como una operación del objeto receptor, manejado por un método del receptor y se implementa como una acción o transición de la máquina de estados.
Paso de tiempo: representa el paso del tiempo (ocurrencia de un tiempo absoluto respecto de un reloj real o virtual o el paso de una cantidad de tiempo dada desde que un objeto entra en un estado). Ejemplo: Palabra clave after, after (2 segundos); after 1 ms desde la salida del evento.
Cambio de estado: evento que representa un cambio en el estado o el cumplimiento de una condición. Ejemplo: Palabra clave when, seguida de una expresión booleana, que puede ser de tiempo o de otra clase: when (hora = 11:30); when (altitud < 1000).

TRANSICIONES I

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 signatura de una transición es como sigue: Evento(argumentos) [condición] / accion

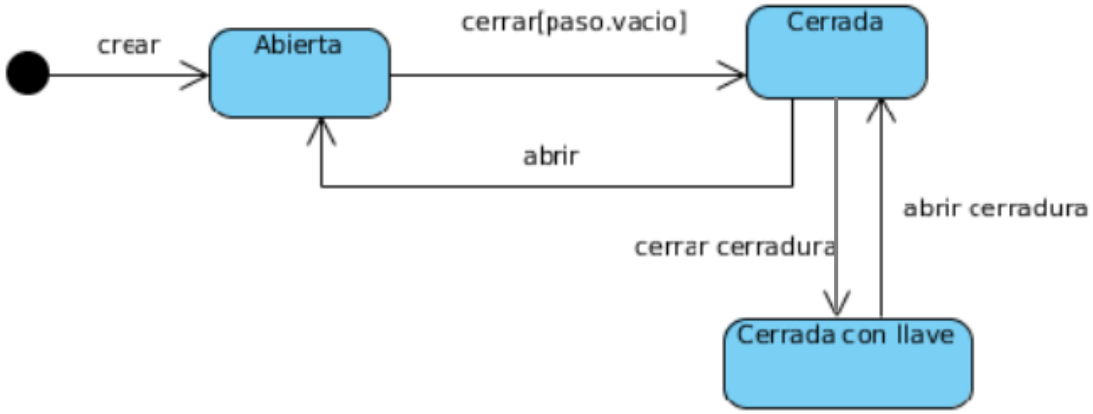
Elementos de una transición: (Todos los elementos son opcionales)
Estados origen y destino: la transición se disparará si, estando en el estado origen se produce el evento de disparo y se cumple la condición de guarda (si la hay), pasando a ser activo el estado final.
Evento de disparo: cuando se produce un evento, afecta a todas las transiciones que lo contienen en su etiqueta. Todas las apariciones de un evento en la misma máquina de estados deben tener la misma signatura. Los tipos de evento los hemos visto en el punto anterior.
Condición de guarda: expresión booleana. Si es falsa, la transición no se dispara, y si no hay otra transición etiquetada con el mismo evento que pueda dispararse, éste se pierde.
Acción: computación atómica ejecutable. Puede incluir llamadas a operaciones del objeto que incluye la máquina de estados (o sobre otros visibles), creación o destrucción de objetos o el envío de una señal a otro objeto.

TRANSICIONES II

Recordemos el diagrama de estado de la puerta: ¿Qué significa la signatura de la transición "cerrar [paso.vacio]"?

Que para cerrar la puerta el paso debe estar vacío.

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.



6. DIAGRAMAS DE ACTIVIDAD I

El Diagrama de Actividad es una especialización del Diagrama de Estado, organizado respecto de las acciones, que se compone de una serie de actividades y representa cómo 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 actividad. 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. En concreto define la lógica de control:

- **En el modelado de los procesos del negocio:** Permiten especificar y evaluar el flujo de trabajo de los procesos de negocio.
- **En el análisis de un caso de uso:** Permiten comprender qué acciones deben ocurrir y cuáles son las dependencias de comportamiento.
- **En la comprensión del flujo de trabajo, a través de varios casos de uso:** Permiten representar con claridad las relaciones de flujo de trabajo (workflow) entre varios casos de uso.
- Aclaran cuando se trata de expresar **aplicaciones multihilos.**

6. DIAGRAMAS DE ACTIVIDAD II

Un diagrama de actividad 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. El grafo parte de un nodo inicial que representa el estado inicial y termina en un nodo final.

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.

ELEMENTOS DEL DIAGRAMA DE ACTIVIDAD I

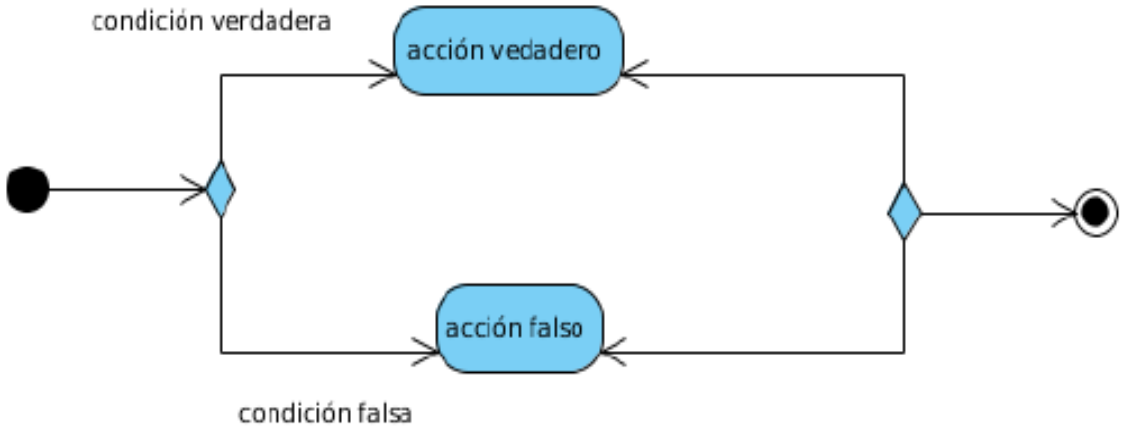
Normalmente los diagramas de actividades contienen: estados de actividad y estados de acción.

- **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.

ELEMENTOS DEL DIAGRAMA DE ACTIVIDAD II

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 (Descision 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.

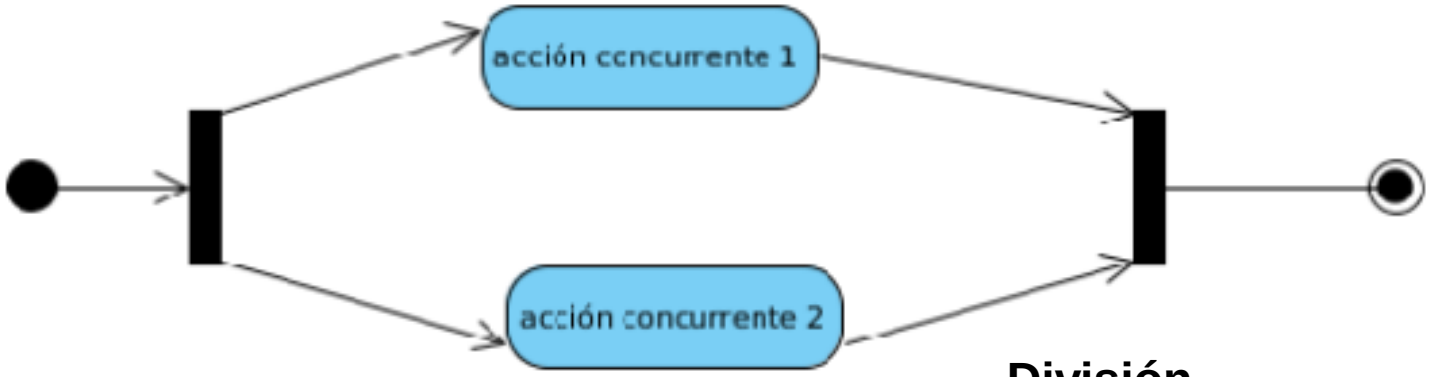


Bifurcación

ELEMENTOS DEL DIAGRAMA DE ACTIVIDAD III

Transiciones: (seguimos)

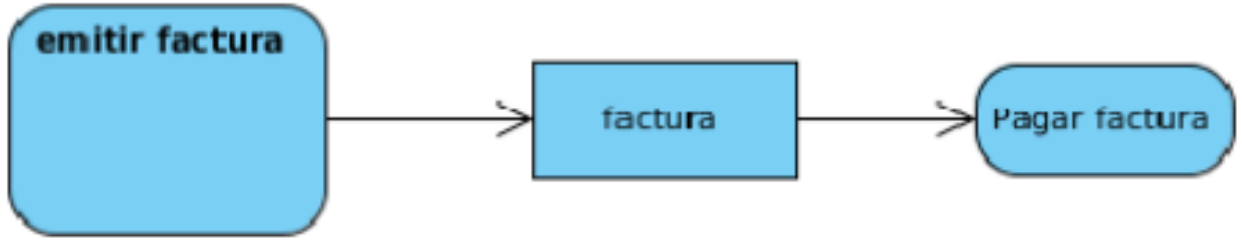
- Fusión (Merge node): Redirigen varios flujos de entrada en un único flujo de salida. No tiene tiempo de espera ni sincronización.
- División (Fork node): Permiten expresar la sincronización o ejecución paralela de actividades. Las actividades invocadas después de una división son concurrentes.
- Unión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



División

ELEMENTOS DEL DIAGRAMA DE ACTIVIDAD III

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.

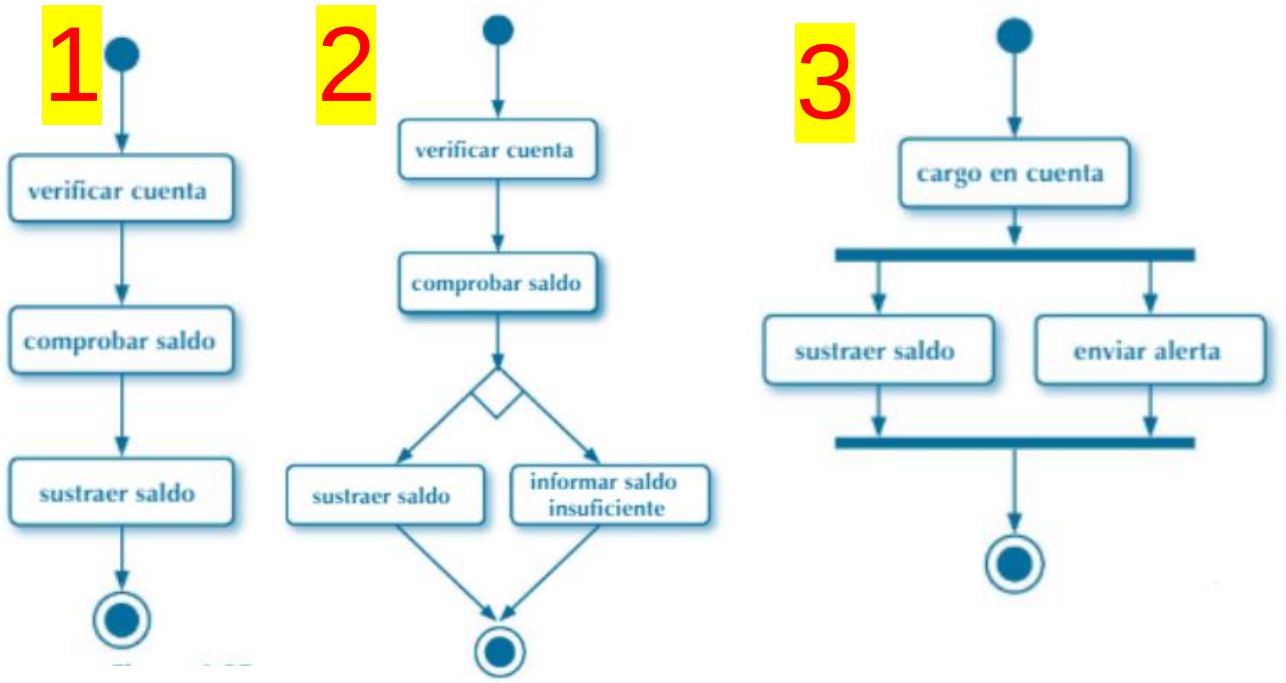
EJEMPLO.

En un banco queremos modelar la actividad de retirada de efectivo. Este proceso se divide en tres actividades.

1. Verificar que la cuenta existe y el usuario que quiere retirar es propietario o autorizado.
2. Comprobar el saldo de la cuenta que sea superior al importe retirado.
3. Retirar el dinero de la cuenta.(1)

Pero, ¿si no hay suficiente saldo? Hay que tomar decisiones.(2)

En ocasiones en sistemas OO, podemos plantear procesos concurrentes. Cuando se realiza un cargo a una cuenta de cualquier cliente del banco, se envía una alerta al móvil, ambas operaciones se realizan en concurrencia.



EJERCICIO 3

- Realizar la práctica del AULA VIRTUAL: P03-Explicación del Diagrama de Comportamiento-casos de uso. **Gestión de fincas e inmuebles**.

  - Escribe una documento con un esquema o mapa conceptual (en una página). Lo mas gráfico y visual posible (figuras). Deben quedar descritas todas las posibilidades en cuanto a ELEMENTOS y RELACIONES.
    1. Lee y trata de entender el ejercicio y como está razonado.
    2. Haz tu mapa conceptual indicando los pasos para realizar un diagrama de caso y utiliza el ejercicio de Gestión de fincas para extraer los ejemplos que ilustren cada caso.
  - Súbelo al aula virtual (AV).

EJERCICIO 4

- Realizar la práctica del AULA VIRTUAL: P04-Diagrama de Comportamiento-casos de uso. **Gestión calificaciones.** Para realizarlo utiliza la herramienta DIA o alguna similar que sea gratuita.

Escribe una documento con portada, índice, url consultadas, etc.

- Debes incluir el enunciado de cada ejercicio, razonando como llegas a obtener el diagrama, pon el diagrama en el documento como una imagen.
- No es necesario que expliques como usas la herramienta DIA o similar.
- Súbelo al aula virtual (AV).

EJERCICIO 5

- Realizar la práctica del AULA VIRTUAL: P05-Diagrama de Comportamiento-casos de uso. **Puntos de información**. Para realizarlo utiliza la herramienta DIA o alguna similar que sea gratuita.

Escribe una documento con portada, índice, url consultadas, etc.

- Debes incluir el enunciado de cada ejercicio, razonando como llegas a obtener el diagrama, pon el diagrama en el documento como una imagen.
- No es necesario que expliques como usas la herramienta DIA o similar.
- Súbelo al aula virtual (AV).