

## Diagramas Comportamiento UML

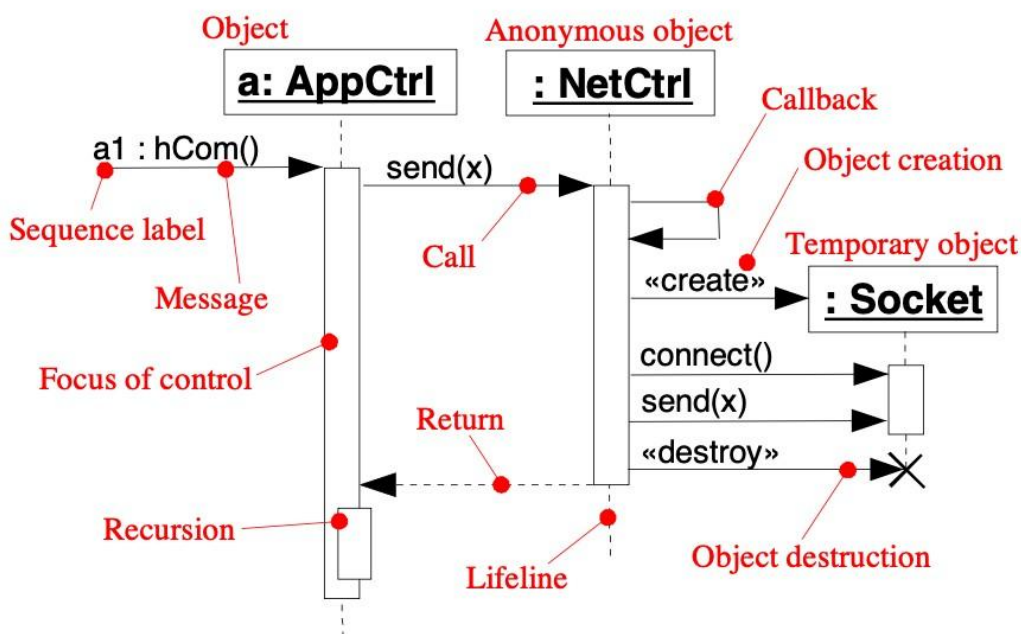
### 1. Diagramas de interacción

Los diagramas de interacción sirven para representar la comunicación entre objetos. Se enfocan en representar los mensajes que se envían entre objetos y como estos mensajes van juntos para lograr la funcionalidad. Mientras los diagramas estructurales muestran qué objetos van juntos para lograr un requerimiento en particular, los diagramas de interacción muestran exactamente cómo estos objetos lo realizan. Existen dos tipos de diagramas de interacción: los diagramas de secuencia y los diagramas de comunicación.

### 2. Diagrama de Secuencia

El diagrama de secuencia se centra en el intercambio de mensajes entre líneas de vida (objetos). Este diagrama es como un “video” (dinámico) con secuencia de acciones entre distintos colaboradores. Muestra el flujo de mensajes entre los objetos de un sistema para cumplir con una cierta funcionalidad. Permite representar el comportamiento del proceso que describe esa funcionalidad.

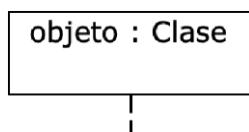
Los participantes involucrados son: actor, objetos, componente software, hardware. En el diagrama se ven reflejado los mensajes entre participantes. Se muestran los mensajes entre participantes ordenados secuencialmente, y podemos definir bloques (if, while, do/while). Actualmente estos diagramas se usan poco.



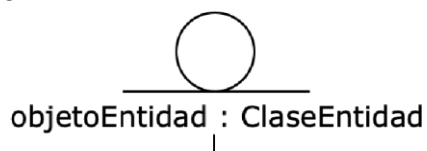
## Objetos

A los objetos los podemos representar de distintas formas: con una caja o con un estereotipo.. Tenemos tres estereotipos que diferencian los tipos de clases: entidad o modelo, vista y controlador. No importa cual se use, cuando lo nombramos un objeto, ponemos primero el nombre del objeto y luego dos puntos y el nombre de la clase. Si el objeto no tiene nombre se deja vacío y solo se coloca dos puntos y el nombre de la clase. La representación UML de esto es:

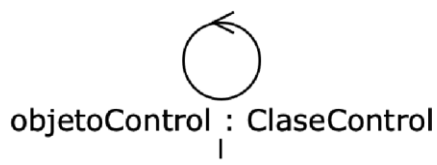
- Objeto cualquiera:



- Objeto de clase Entidad o Modelo:



- Objeto de clase Control:

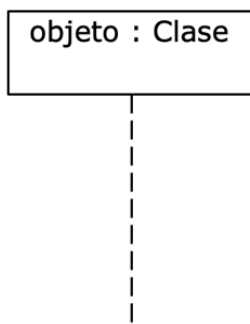


- Objeto de clase Vista:



## Línea de vida

La línea de vida del objeto representa la existencia del objeto. Indica que el objeto existe. Se dibuja como una línea punteada. Su representación UML:



## Mensajes

Los mensajes representan la comunicación entre objetos. Tenemos varios tipos de mensajes:

- **Mensajes sincrónicos:** se envía un mensaje y la acción queda paralizada hasta recibir respuesta

Representación UML: 

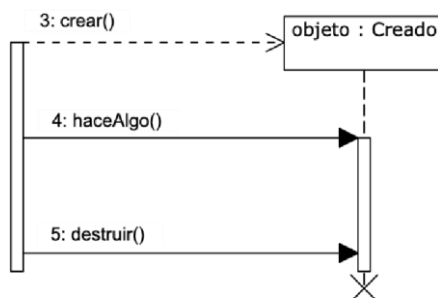
Notar que la punta esta rellena de negro

- **Mensajes asincrónicos:** en el contexto de nuestro curso no usamos estos mensajes, no nos interesan
- **Mensajes de retorno:** representan las respuestas a mensajes

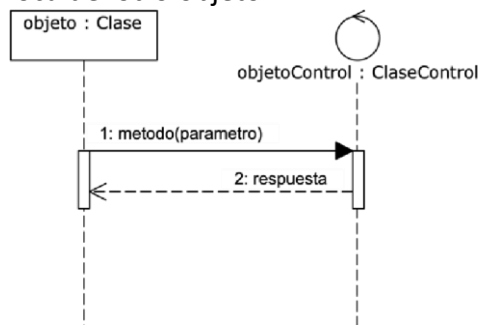
Representación UML: 

Cuando usemos estos mensajes se coloca como texto descriptivo del mensaje solo lo que esta retornando, no colocar palabras del tipo “se retorna edad”, solo colocar “edad”

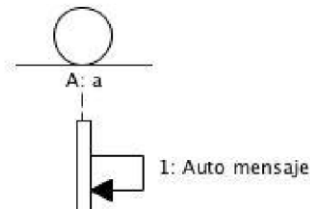
- **Mensajes de creación y destrucción:** cuando se envian estos mensaje se crea un objeto o destruye un objeto. Representación UML:



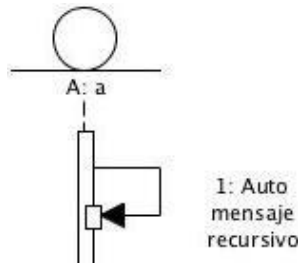
- **Notación para mensaje:** los mensajes se identifican con un número secuencias y un texto descriptivo que va sobre la línea del mensaje y alineado hacia el lugar del objeto que envía el mensaje. Este texto es el nombre del método que se invoca del otro objeto.



- **Automensajes:** un objeto puede enviarse mensajes a si mismo. La notación UML sería:



- **Mensajes recursivos:** un objeto puede mandarse un mensaje y crear otra caja de activación. Se usa cuando sobre el mensaje existe, por ejemplo el operador new que esta a otro nivel dentro de la línea de vida del objeto. La notación UML sería:



### Caja de activación

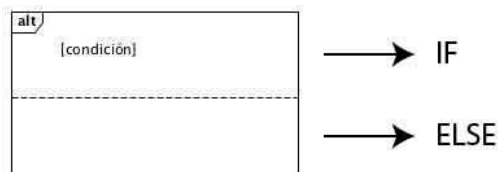
Cuando un mensaje es recibido, una actividad se inicia en el objeto que lo recibe, a esto se le denomina caja de activación. Esto representa que se activa el objeto y que hace una serie de actividades antes de retornar el control. Representación UML:



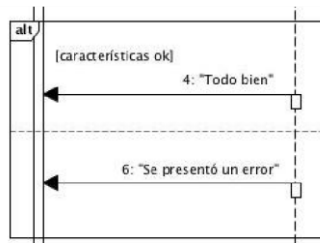
### Fragmentos combinados

Los fragmentos combinados sirven para representar alternativas. Tenemos cuatro tipos: Alt, Opt, Loop y SD.

- **ALT:** representa una alternativa (if ... then ... else). Se usa para agrupar varios mensajes condicionados, con un sí no. Representación UML:



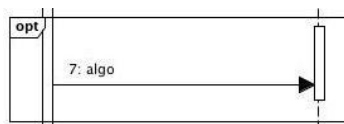
Note que la condición del ELSE no se coloca, esta sobre entendida. Ejemplo:



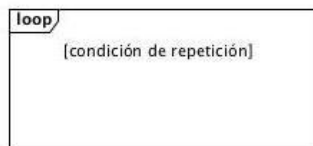
- **OPT:** representa una opción (case) que puede o no ocurrir. Las opciones son cosas que hacemos de manera independiente. Se usa para agrupar varios mensajes condicionales que se hacen de manera independiente. Representación UML:



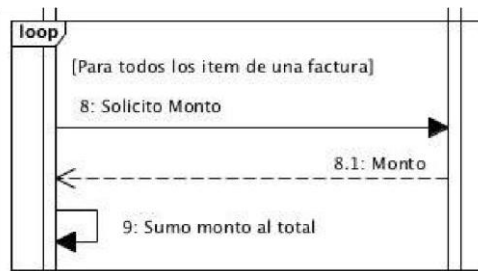
Ejemplo:



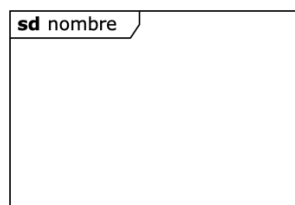
- **LOOP:** representa cosas que se repiten (ciclos). Es como un for o while en el código. Representación UML:



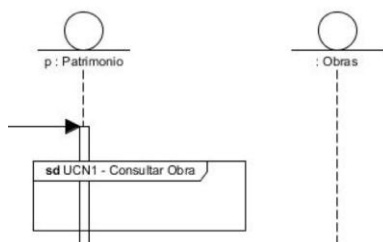
Ejemplo:



- **SD:** se usan para encerrar un diagrama de secuencia entero. Representa los *include* y *extend* entre Casos de Uso en un diagrama de secuencia. Representación UML:

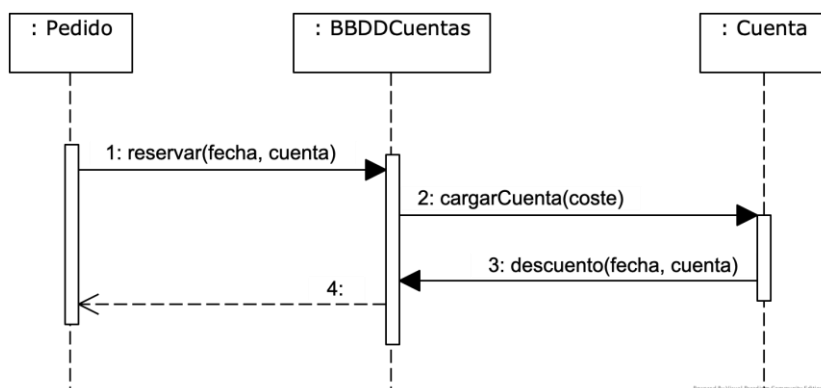


Ejemplo:



### Ejemplo diagrama secuencia

Estos diagramas se realiza a partir de la descripción de un Caso de Uso. Un ejemplo de procesar pedido:

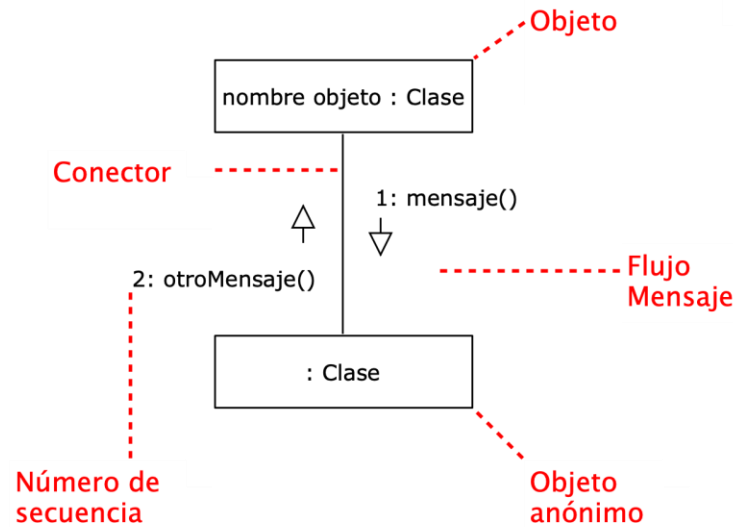


## 3. Diagrama de Comunicación

Un diagrama de comunicación se enfoca en la interacción entre líneas de vida donde la arquitectura de la estructura interna y cómo esto se corresponde con el paso del mensaje es fundamental. La secuencia de mensajes se da a través de una numeración.

Este diagrama es como una "foto movida" (estática) sin secuencia de acciones entre colaboradores. Muestra el paso de mensajes de unos componentes a otros en los que se indican sus funcionalidades o métodos. Permite modelar diferentes situaciones en casos de colaboración entre varios objetos, extendiendo los posibles casos de uso

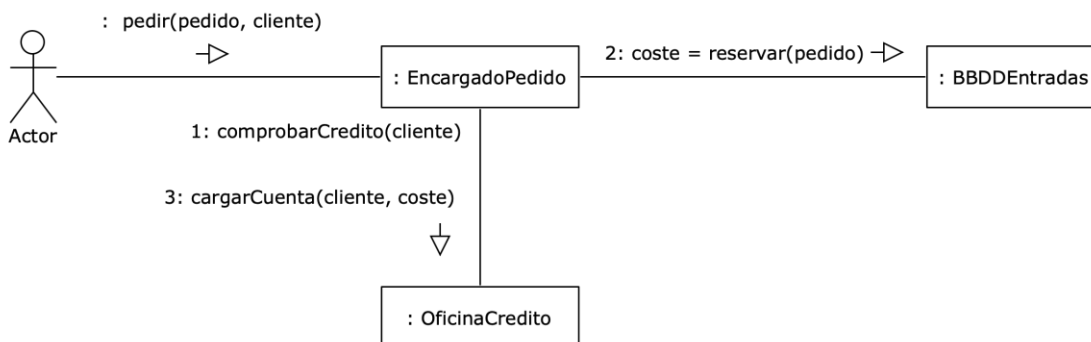
Los participantes involucrados en este diagrama son los: actor, objetos, componente software, hardware. En el diagrama se ven reflejado los mensajes entre participantes colaboradores. Este diagrama enfatiza la organización estructural de los objetos que envían y reciben mensajes. Este diagrama muestra los mensajes que se envían como respuesta a la invocación de un método. La secuencia de estos mensajes se traduce en una serie de sentencias en la definición del método.



Powered By Visual Paradigm Community Edition

Un diagrama de comunicación modela los objetos y enlaces involucrados en la implementación de una interacción e ignora al resto.

Los mensajes entre roles se muestran mediante flechas etiquetadas vinculadas a los conectores. Cada mensaje tiene un número de secuencia, una condición de guarda opcional, un nombre y una lista de argumentos, y un nombre opcional de valor de retorno. El número de secuencia incluye opcionalmente el nombre del hilo. Todos los mensajes en el mismo hilo están ordenados secuencialmente. Los mensajes en diferentes hilos son concurrentes a menos que haya una dependencia explícita en la secuenciación. Se pueden añadir diversos detalles de implementación, como la distinción entre mensajes síncronos y asíncronos. Un ejemplo de tomar pedido:



Powered By Visual Paradigm Community Edition

En la versión 1.0 de UML este diagrama se llamaba diagrama de colaboración.

## 4. Diagrama de Secuencia vs diagrama de Comunicación

Tanto los diagramas de comunicación, como los diagramas de secuencia, muestran interacciones, pero cada uno hace énfasis en aspectos distintos. Los diagramas de secuencia muestran claramente secuencias temporales, pero no muestran explícitamente relaciones entre objetos. Los diagramas de comunicación muestran relaciones entre objetos con claridad, pero las secuencias temporales se obtienen de los números

de secuencia. A menudo, los diagramas de secuencia son más útiles para mostrar el diseño detallado de procedimientos. Sin embargo, una vez que se ha definido la estructura de un procedimiento, los diagramas de comunicación pueden ser más útiles para planificar los pequeños detalles del control.

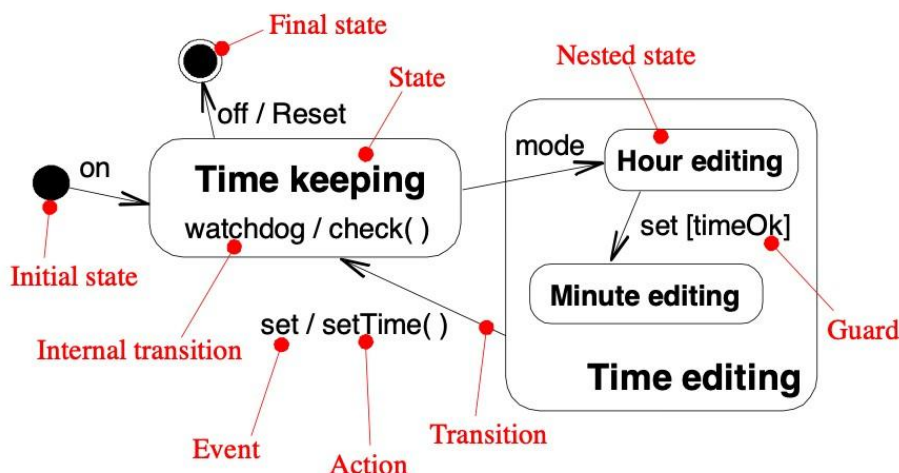
Hay una gran similitud formal entre los diagramas de comunicación y los diagramas de secuencia, pero no son tan similares en la práctica. Los diagramas de comunicación están centrados en describir los mensajes y las colaboraciones. Los diagramas de secuencia, por su particular forma de dibujarse, tiene como objetivo describir la interacción entre objetos en un particular caso de uso.

## 5. Diagrama de Estados

Un diagrama de estados se utiliza para modelar el comportamiento discreto a través de transiciones de estados finitos. Además de expresar el comportamiento de una parte del sistema, las máquinas de estado también se pueden usar para expresar el protocolo de uso de parte de un sistema.

Este diagrama es como un "video" (dinámico) destacando los estados intermedios con secuencia de acciones no destacadas. Se utiliza para describir el comportamiento de un sistema, o de una parte de él, a lo largo de los estados por los que transitan algunos de sus componentes. Un componente se encuentra en un estado hasta que se cumple, o se deja de cumplir una determinada condición; en ese momento cambia de estado. La representación de la transición de un estado a otro es lo que se muestra en un diagrama de estados.

Este diagrama es un grafo con nodos y enlaces. Los nodos son: estados que no tiene subestados (unidad); disyunciones con condiciones (condición evaluada en tiempo de ejecución para determinar el enlace que se puede atravesar); y conjunciones (para la unión de enlaces y continuar por un camino común). Los enlaces por su parte son para el control de flujo de transición de estados por la ejecución de las acciones de las transiciones y evaluación de las condiciones.



En el diagrama de estado se organizan los casos de uso, por eso tienen que tener el nombre del caso de uso, y hay que definir los estados. Este diagrama me explica qué casos de uso puedo realizar dependiendo de los casos de uso hechos previamente. Representación UML general de este diagrama:





## Estados

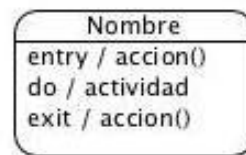
El estado es un resultado de la ejecución de una actividad previa hecha por el Objeto y esta determinado por el valor de sus atributos y enlaces a otros objetos. Un estado es la abstracción de los valores de los atributos de un objeto y de los enlaces con otros objetos en un determinado momento. Por ejemplo, el vehículo (objeto) esta detenido (estado). Todos los objetos tienen estados. Un objeto permanece en un estado durante una cantidad de tiempo finito. Hay tres tipos de Estados:



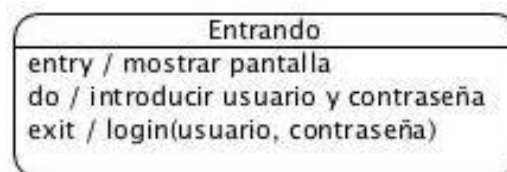
El estado intermedio puede tener:

- **Nombre:** normalmente en gerundio
- **Entry / accion():** acciones que se van a realizar a la entrada de este estado (no consume casi tiempo)
- **do / actividad:** actividades que se van hacer en ese estado (consume tiempo)
- **Exit / accion():** acciones que se hacen a la salida del estado (no consume casi tiempo)

Notación UML:



Por ejemplo:



## Subestados

Un estado puede tener subestados. El estado internamente tiene sus subestados que pueden mostrar su propio diagrama de estado

## Acciones

Operaciones atómicas, que no consumen mucho tiempo, se consideran procesos casi instantáneos. No pueden ser interrumpidas. Pueden definirse dentro de los estados o estar asociadas a las transiciones.

## Actividades

Operaciones que toman tiempo para completarse. Pueden ser interrumpidas. Se asocian con un estado. Una actividad puede involucrar un secuencia de acciones.

## Evento

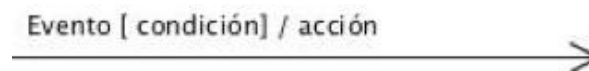
Es la ocurrencia de algo en un punto del tiempo y se considera que no tiene duración. Un objeto cambia de un estado a otro cuando algo pasa, esto es llamado “evento”. La respuesta a un evento depende del estado en que se encuentra el objeto.

## Transiciones

Una transición es el cambio de estado causado por un evento. Por ejemplo, cuando se enciende el vehículo. Cuando un evento ocurre, se realiza la transición de un estado a otro.

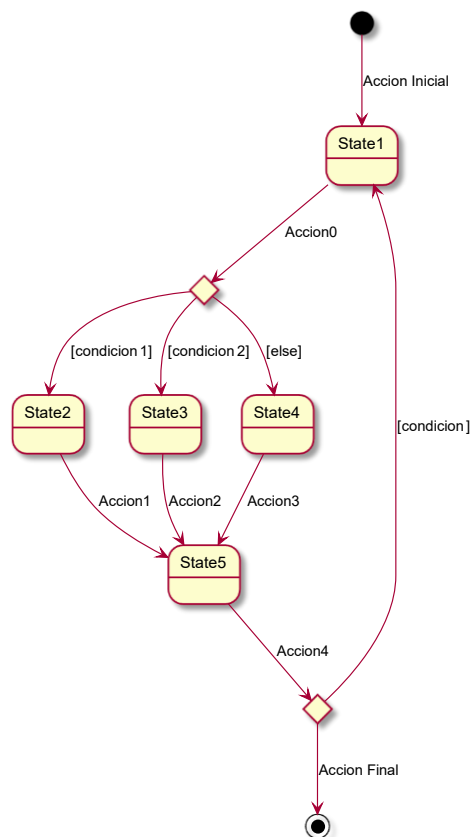
¿El evento de transición es el entry de un estado? No, el evento de transición causa el cambio de un estado a otro, y el entry de un estado indica que al llegar a ese estado debe ocurrir esa otra acción.

Representación UML:



El evento ocurre bajo la condición, o sea, si se cumple la condición al generarse el evento, se realiza la acción.

## Ejemplo de diagrama de Estado



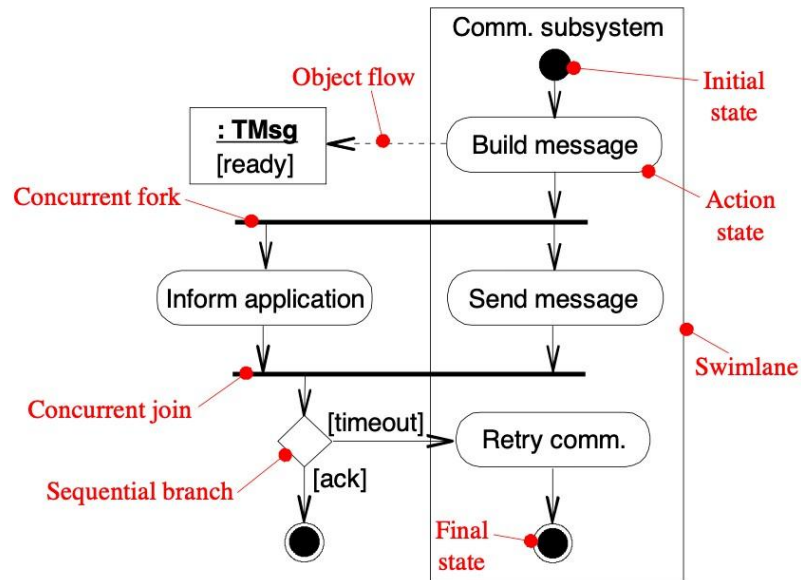
## 6. Diagrama de Actividad

El diagrama de actividad muestra la secuencia y las condiciones para coordinar los comportamientos de nivel inferior, en lugar de los clasificadores que poseen esos comportamientos. Estos son comúnmente llamados modelos de flujo de control y flujo de objetos. Se usa principalmente para modelar el flujo de trabajo.

Este diagrama es como un "video" (dinámico) con secuencia de acciones destacadas. Muestran los procesos, desde su inicio a fin, que modelan los casos de uso del sistema. Se indica la secuencia por la que transita un paquete de información, desde el nodo inicial al nodo final. Muestran las actividades y flujos de control del proceso.

Este diagrama es un grafo con nodos y enlaces. Los nodos son: acciones (que representa un solo paso atómico o unidad dentro de la actividad, es decir, que no se descompone más dentro de la actividad); disyunciones (condición evaluada en tiempo de ejecución para determinar el enlace que se puede atravesar) y conjunciones (para la unión de enlaces y continuar por un camino común).

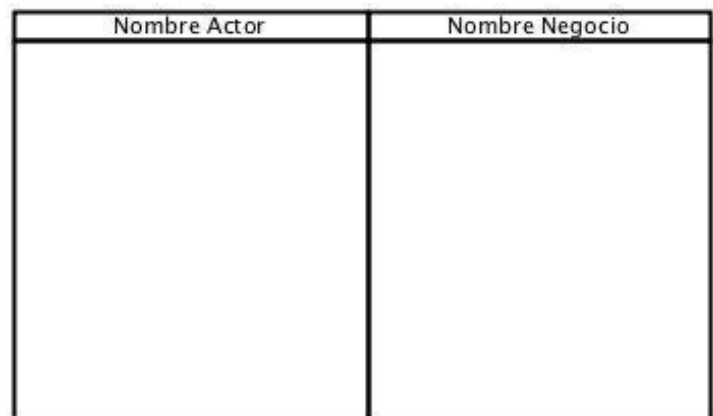
Los enlaces por su parte son para el control de flujo de ejecución de las acciones y evaluación de las condiciones. Actualmente se usan poco estos diagramas.



## Particiones

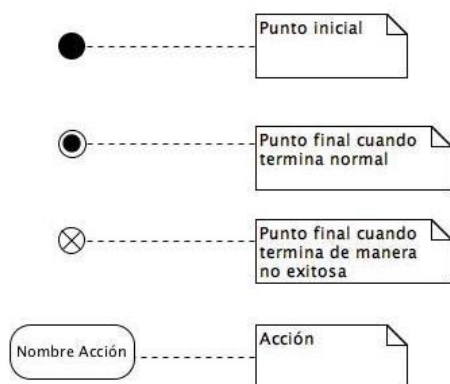
Muestra las acciones y quien o en qué parte de la organización se realizan. Las particiones contienen las acciones o actividades. Se puede representar de manera vertical u horizontal y llevan un nombre. El nombre indica quién realiza las acciones. Si no tengo un trabajador del negocio claro, uso el nombre del negocio

Notación UML:



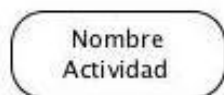
## Acciones

Una acción es algo que se realiza para producir un resultado. Tenemos cuatro tipos: punto de inicio, punto de final cuando termina normal, punto de final cuando termina de manera no exitosa (anormal) y acción. Notación UML:



## Actividades

Las actividades representan una o más acciones. Gráficamente son iguales a las acciones, se diferencian por el nombre. Notación UML:



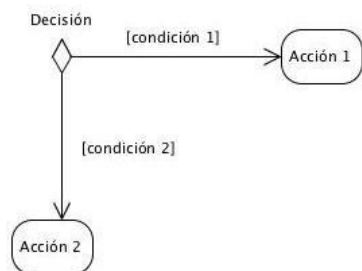
## Transiciones

Las transiciones representan un flujo de control. Conectan dos acciones o actividades. Llevan sentido de navegabilidad para saber desde donde voy y a donde avanzo. Notación UML:

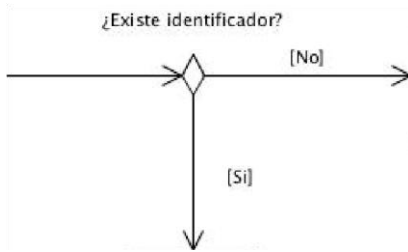


## Decisiones y condiciones

Las decisiones y condiciones se usan para representar bifurcaciones en base a una toma de decisión. Notación UML:

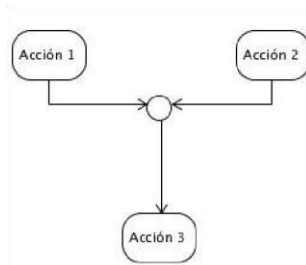


Ejemplo:



## 1. Conectores

Se usan para que una acción tenga una sola transición de entrada. Notación UML:

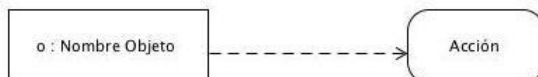


## 2. Objetos

Representan objetos que son entradas y/o salidas de las acciones o actividades. Los objetos son las entidades del negocio o sistema. Notación UML:



- Si un objeto es requerido por una acción se representa como:



- Si una acción modifica un objeto se representa como:

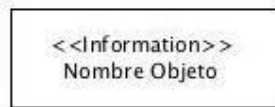


Por ejemplo:



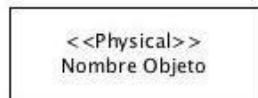
Se diferencian los objetos físicos de los objetos de información. El estereotipo <<Information>> se utiliza para indicar que un objeto es de tipo información.

- Un objeto de este tipo puede ser: Fecha de nacimiento
- No deben existir transiciones entre objetos de información que representen un flujo de control
- Notación UML:



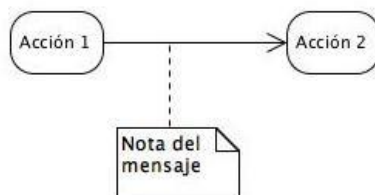
El estereotipo <<Physical>> se utiliza para indicar que un objeto representa un objeto físico real

- Un objeto de este tipo puede ser: HistoriaMédica • Notación UML:

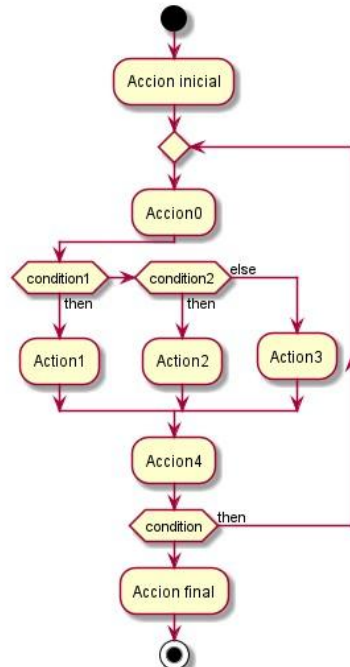


### Notas de mensajes

Se utilizan para indicar que un objeto del negocio se genera en una acción o actividad para un actor del negocio. Notación UML:



### Ejemplo de diagrama de Actividad:



## 7. Diagrama de Actividad vs Diagrama Estado

En el diagrama de actividad, cada caja son acciones, los rombos son decisiones (antiguos diagramas de flujo o organigramas). En el diagrama de estado (autómatas), cada acción es una flecha o transición, y las

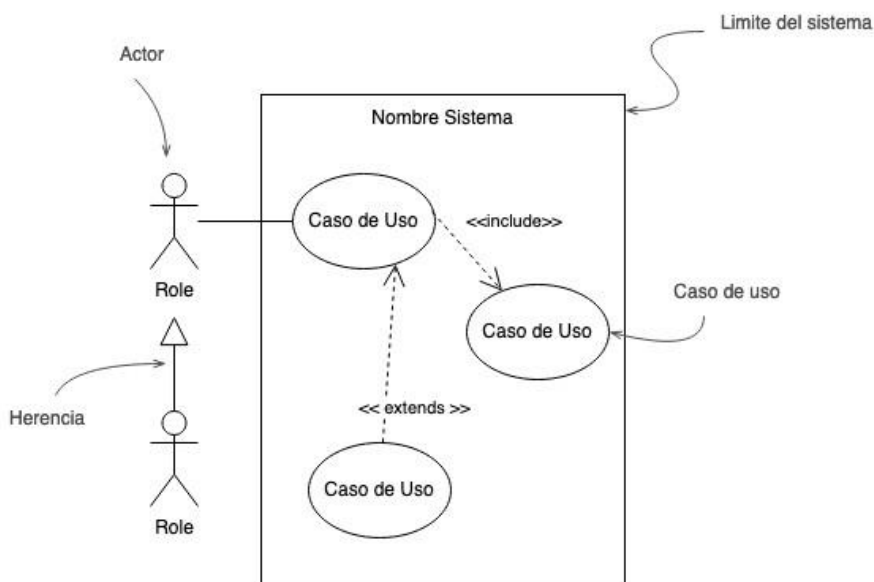
cajas son estados, los rombos son iguales (decisiones). Se puede hacer un diagrama de estado equivalente a un diagrama de actividad y viceversa.

## 8. Diagrama de Casos de Uso

El diagrama de Casos de uso describe un conjunto de acciones (casos de uso) que algunos sistemas o sistemas (sujetos) deben o pueden realizar en colaboración con uno o más usuarios externos del sistema (actores) para proporcionar algunos resultados observables y valiosos a los actores u otros interesados del sistema(s).

Este diagrama es una "Foto" de tipos de "conversaciones" sin secuencia de acciones entre actor y el sistema informático. Este diagrama esta estrechamente ligado con los requisitos funcionales del sistema. Su objetivo es mostrar de manera gráfica la funcionalidad del producto de software que se va a implementar. Se utiliza en la comunicación con el cliente, para explicitar las funcionalidades que se van a implementar. Tiene dos participantes: Casos de Uso y Actores. Muestra los enlaces entre actores y casos de Uso, y entre los mismos casos de uso (herencia, extend e include).

En un mismo diagrama se pueden mostrar las diferentes funcionalidades del sistema y las diferentes acciones que puede ejecutar cada uno de los actores que participan, así como las relaciones entre ellos y los bienes y servicios que aparecen.



Ejemplos:



