

# TEMA 6. MÁQUINAS DE ESTADO

## ANÁLISIS Y DISEÑO DE SISTEMAS DE INFORMACIÓN



Universidad  
Francisco de Vitoria  
**UFV** Madrid

# ÍNDICE

## Tema 6: máquinas de estado.

1. Introducción
2. Estados
3. Transiciones
4. Tipos de eventos
5. Tipos de estados
6. Puntos de entrada y salida

# 1.INTRODUCCIÓN

Cada **objeto** toma un **número finito de estados diferentes** durante su vida.

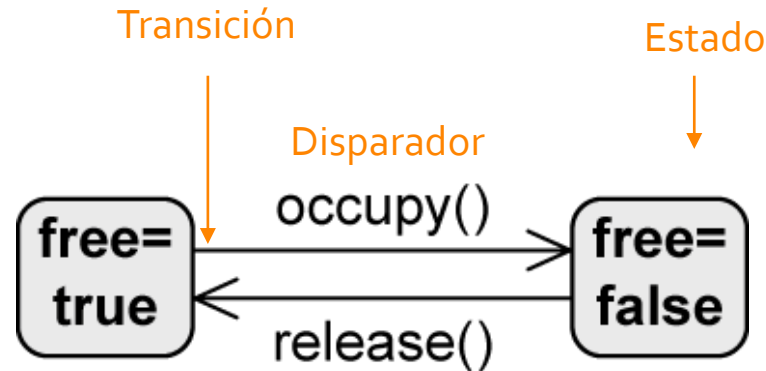
El diagrama de la máquina de estados se utiliza para:

**Diseñar los posibles estados** de un sistema u objeto.

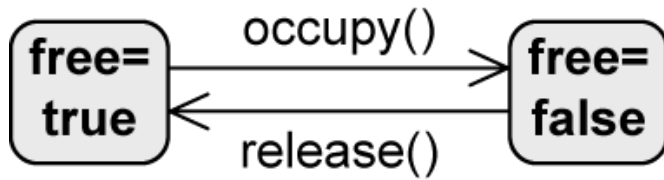
Mostrar **cómo suceden las transiciones de estado** como consecuencia de **eventos**.

Mostrar **qué comportamiento** requiere el sistema u objeto para cada estado.

Ejemplo:

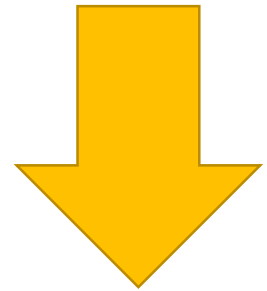
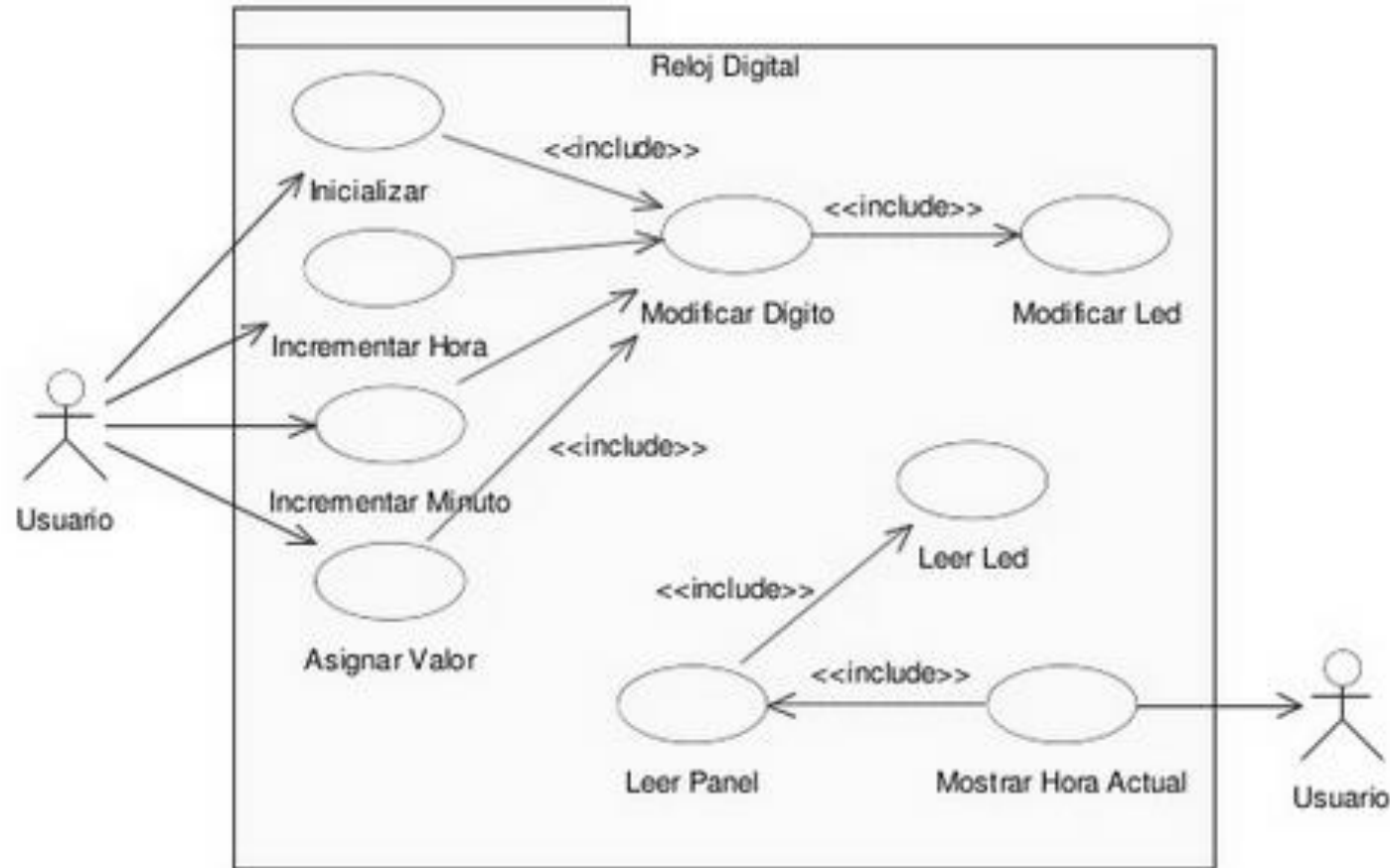


# EJEMPLO: SALA DE LA CONFERENCIA

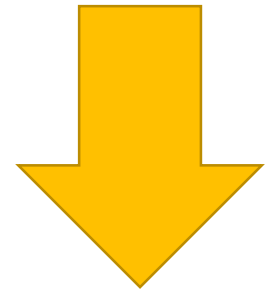
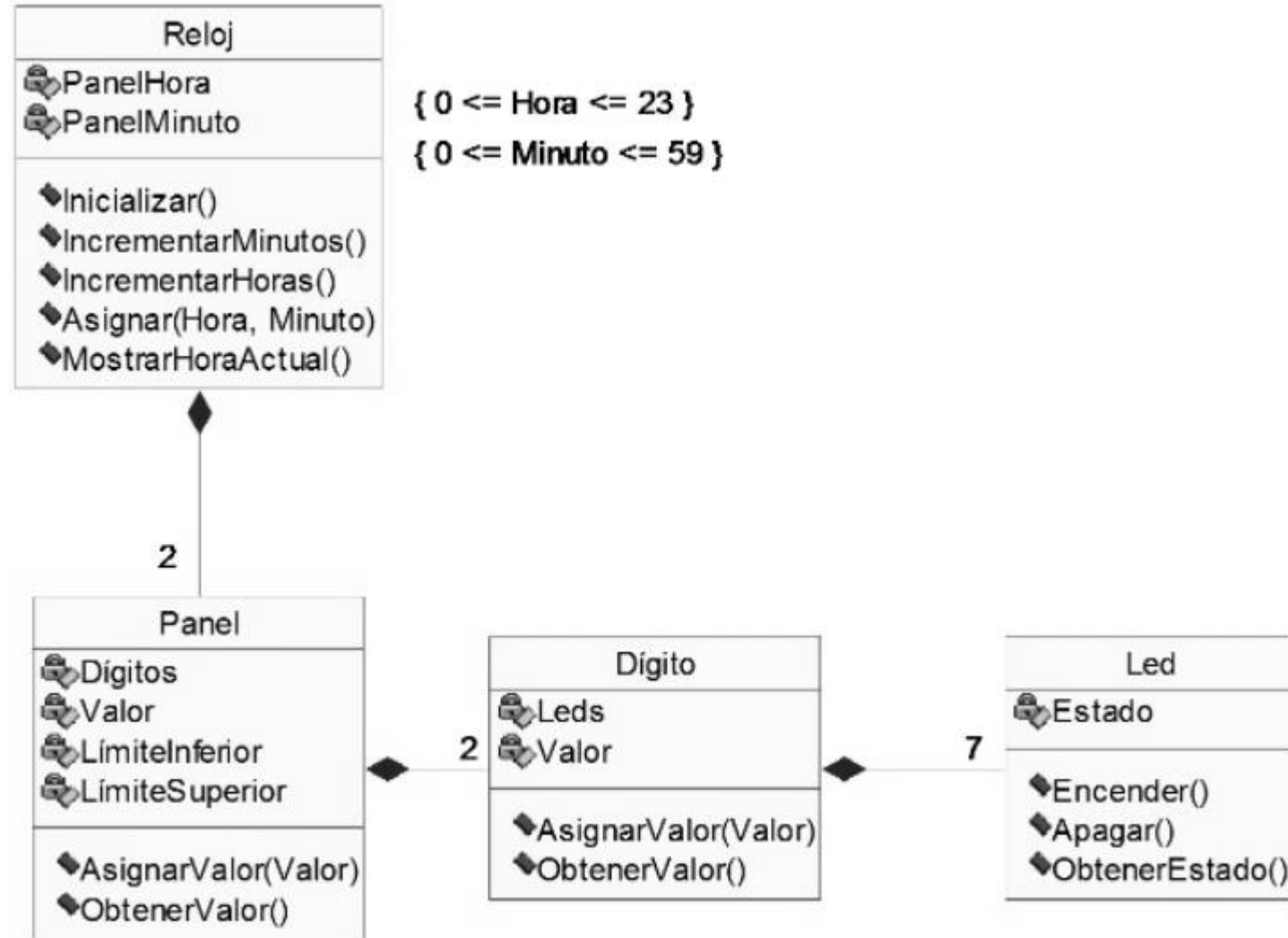


```
class LectureHall {  
    private boolean free;  
  
    public void occupy() {  
        free=false;  
    }  
    public void release() {  
        free=true;  
    }  
}
```

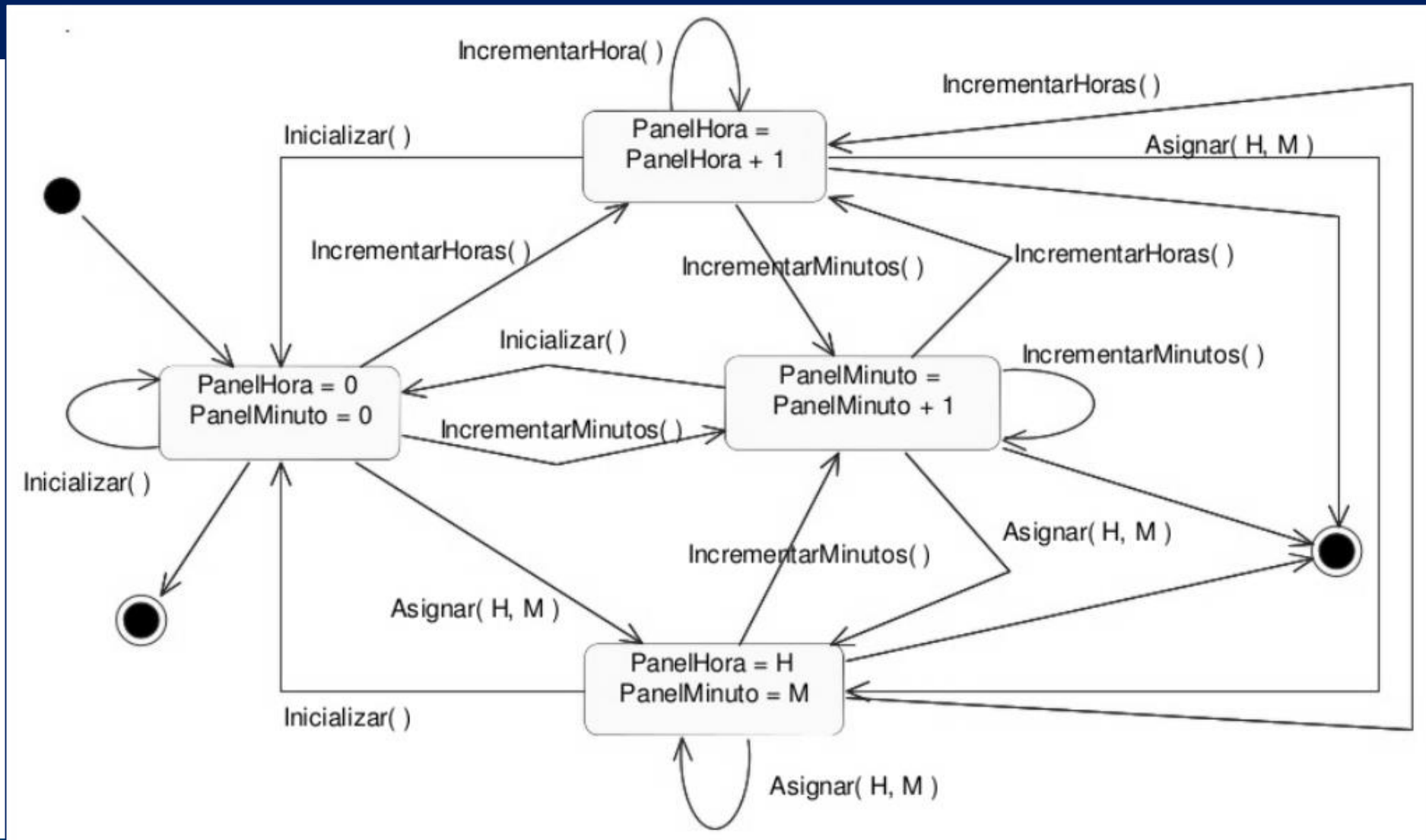
# EJEMPLO: RELOJ DIGITAL



# EJEMPLO: RELOJ DIGITAL

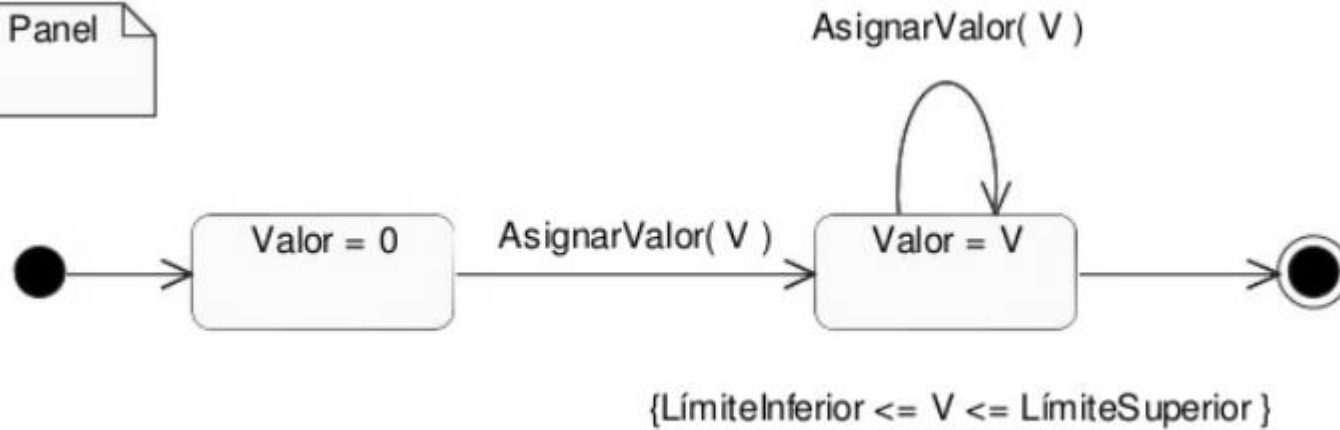


# EJEMPLO: RELOJ DIGITAL

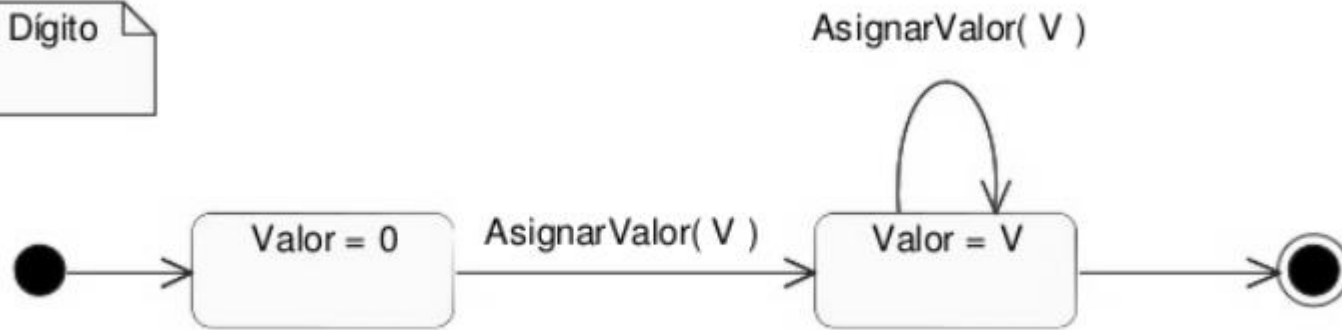


# EJEMPLO: RELOJ DIGITAL

Clase Panel

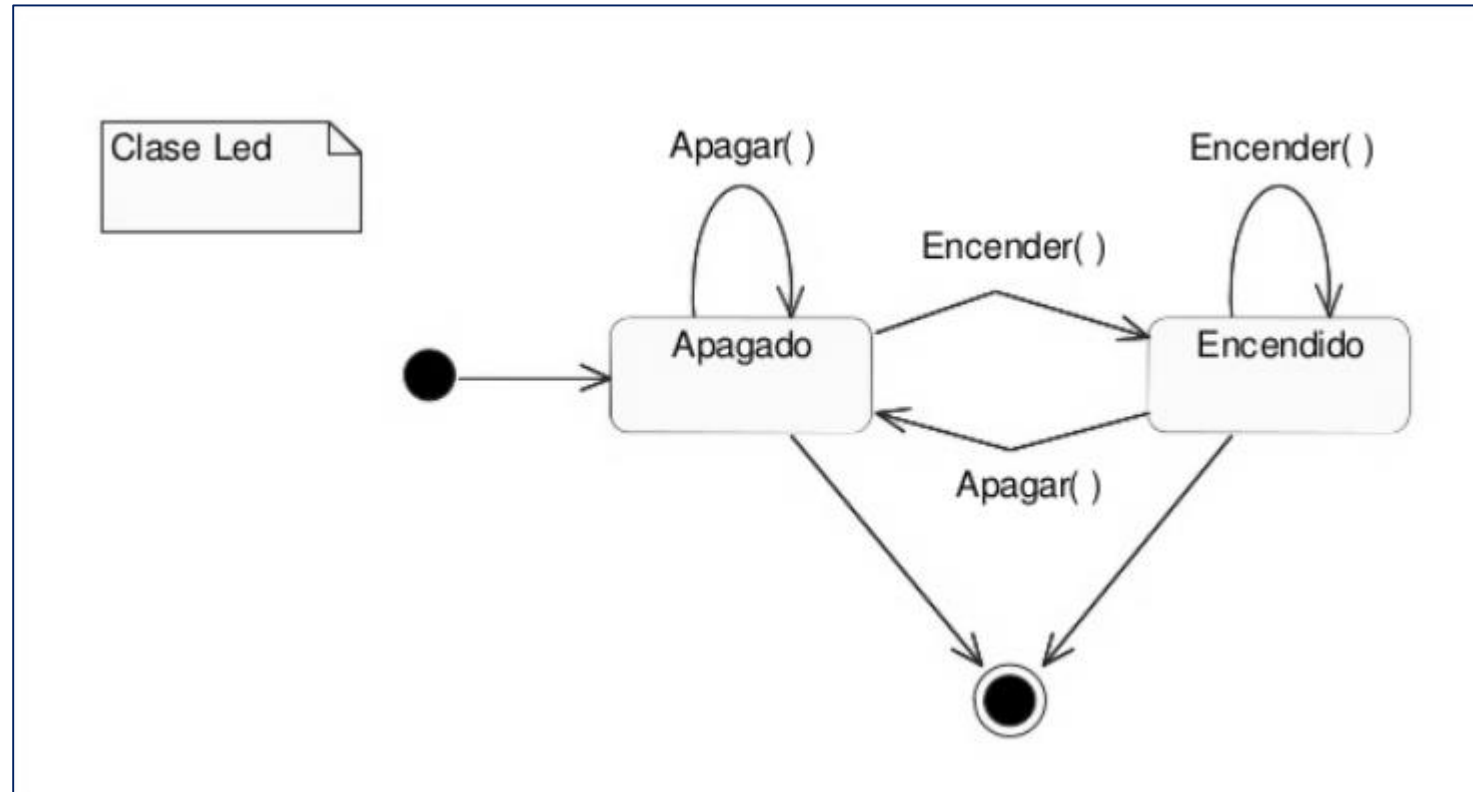


Clase Dígito





# EJEMPLO: RELOJ DIGITAL



## 2. ESTADOS

**Estados** = nodos de la máquina de estados

- Cuando un estado está activo.
- El objeto está en ese estado.
- Todas las actividades internas especificadas en este estado se pueden ejecutar.
- Una actividad puede constar de varias acciones.

entrada / Actividad (...)

Se ejecuta cuando el objeto entra en el estado.

salida / Actividad (...)

Se ejecuta cuando el objeto sale del estado.

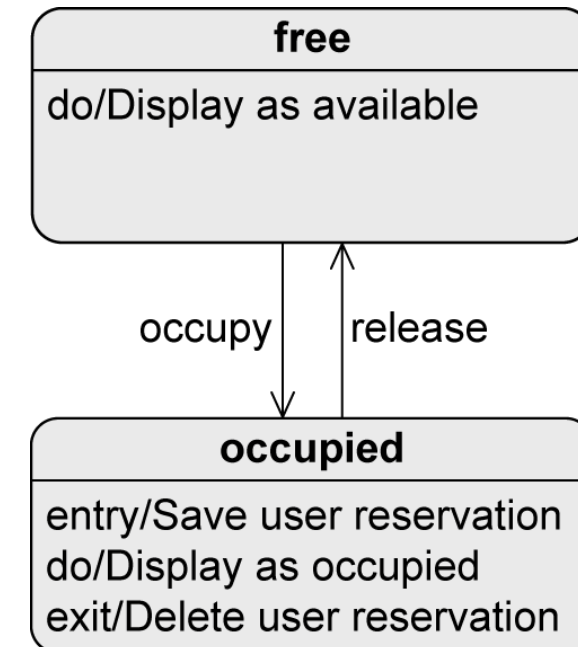
hacer / Actividad (...)

Se ejecuta mientras el objeto permanece en este estado.

**S**

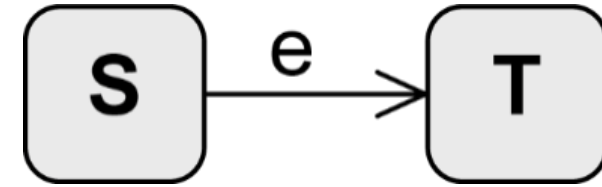
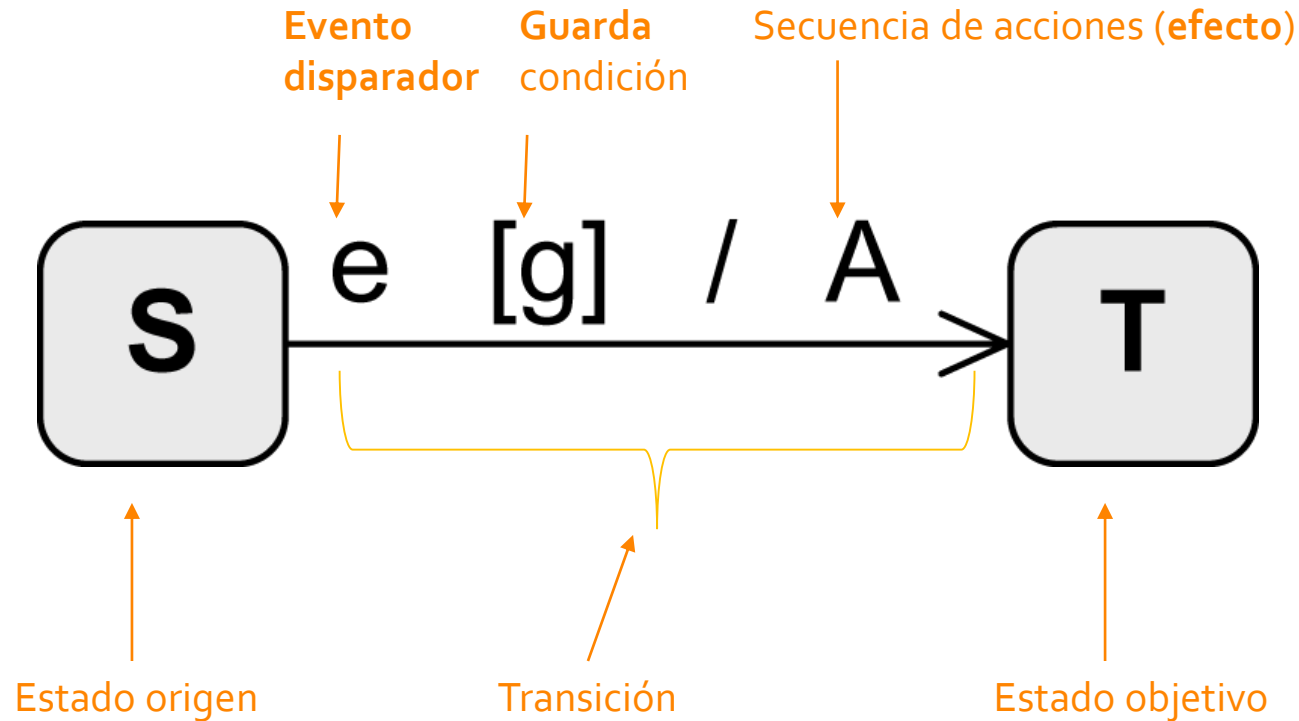
**S**

entry/Activity(...)  
do/Activity(...)  
exit/Activity(...)



# 3. TRANSICIONES

Cambiar de un estado a otro

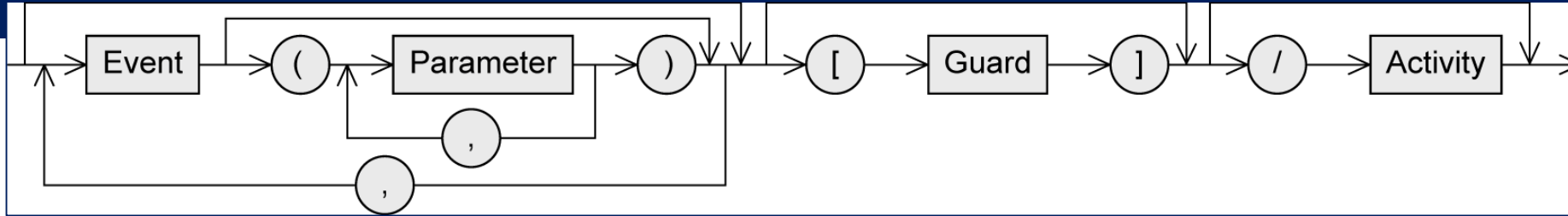


"Trigger" (Disparador) es la causa de la transición, puede ser una señal, un evento, un cambio en alguna condición, o el paso del tiempo.

"Guard" (guarda) es una condición que debe ser verdadera para que el disparador ejecute la transición.

"Effect" (efecto) es una acción que se llama directamente desde el objeto que es el resultado de la transición

# 3. TRANSICIONES



Evento (disparador)

Estímulo exógeno

Puede desencadenar una transición de estado

Guarda (condición)

Expresión booleana

Si el evento ocurre, se revisa la guardia

Si la guardia es verdadera (true)

Todas las actividades en el estado actual se terminan

Se ejecuta cualquier actividad de salida relevante

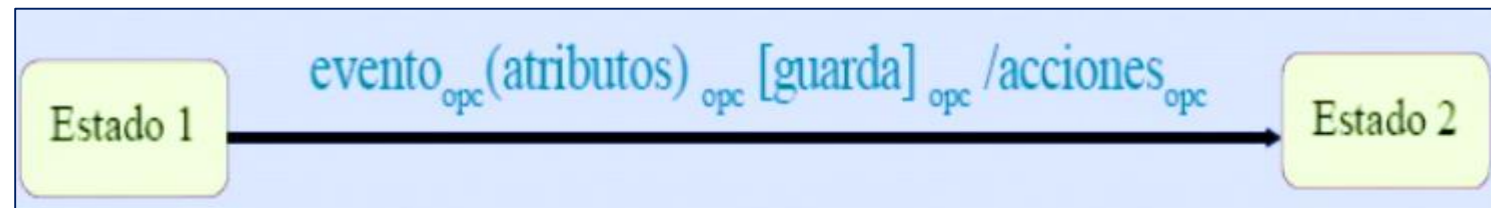
La transición tiene lugar

Si la guardia es falsa (false)

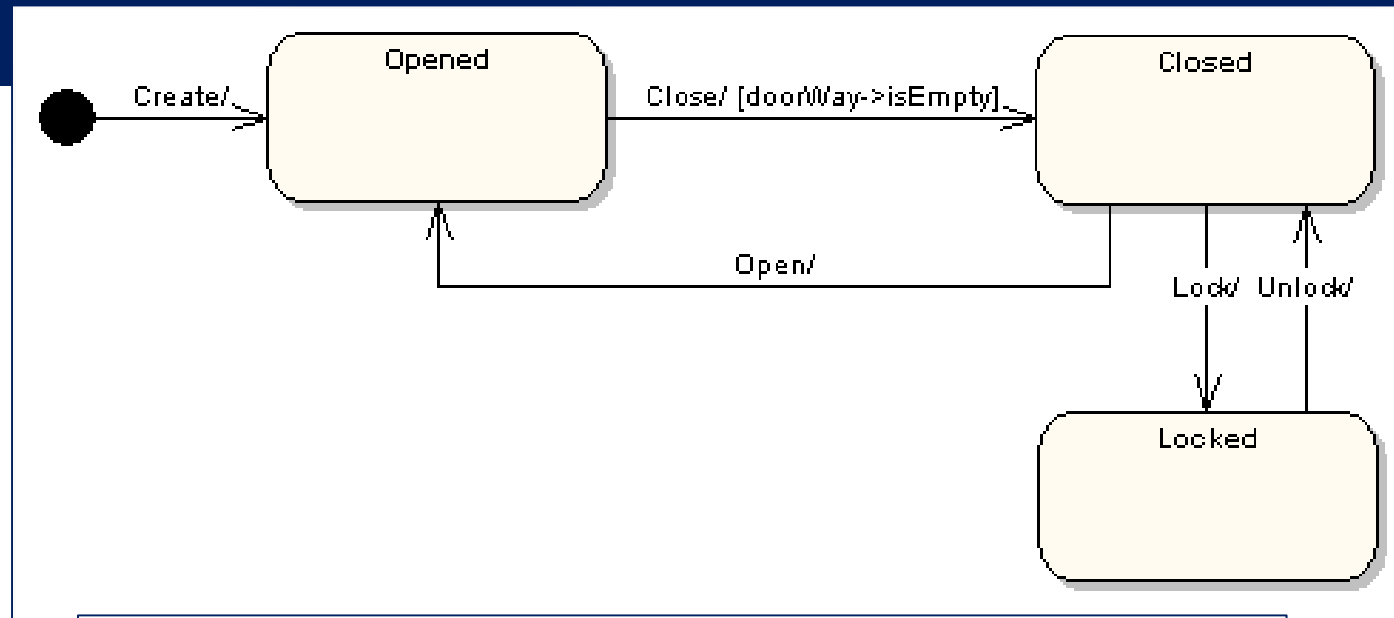
No se produce ninguna transición de estado, el evento se descarta

Actividad (efecto)

Secuencia de acciones ejecutadas durante la transición de estado



### 3. TRANSICIONES



Una **puerta** puede estar en uno de tres estados: "Opened" (**Abierta**), "Closed" (**Cerrada**) o "Locked" (**Bloqueada**). Puede responder a tres estados Abrir, Cerrar, Bloquear y No bloqueado.

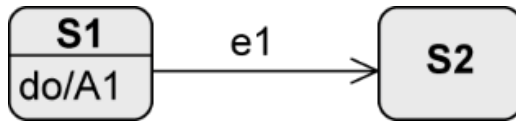
**No todos los eventos son válidos en todos los estados.**

Por ejemplo, si una puerta está abierta, no se puede bloquear hasta que se cierre.

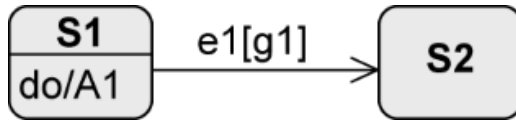
Una transición de estado puede tener una condición de guarda adjunta. Si la puerta está abierta, ésta solo puede responder al evento cerrar si la condición **doorWay->isEmpty** esta completa.

# 3.TRANSICIONES.TIPOS.

¿Cuándo tienen lugar las siguientes transiciones?



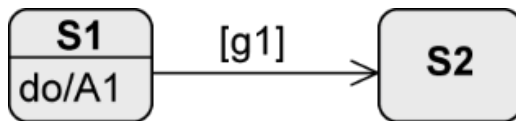
Si ocurre e1, A1 se cancela y el objeto cambia a S2



Si e1 ocurre y g1 se evalúa como verdadero, A1 se cancela y el objeto cambia a S2

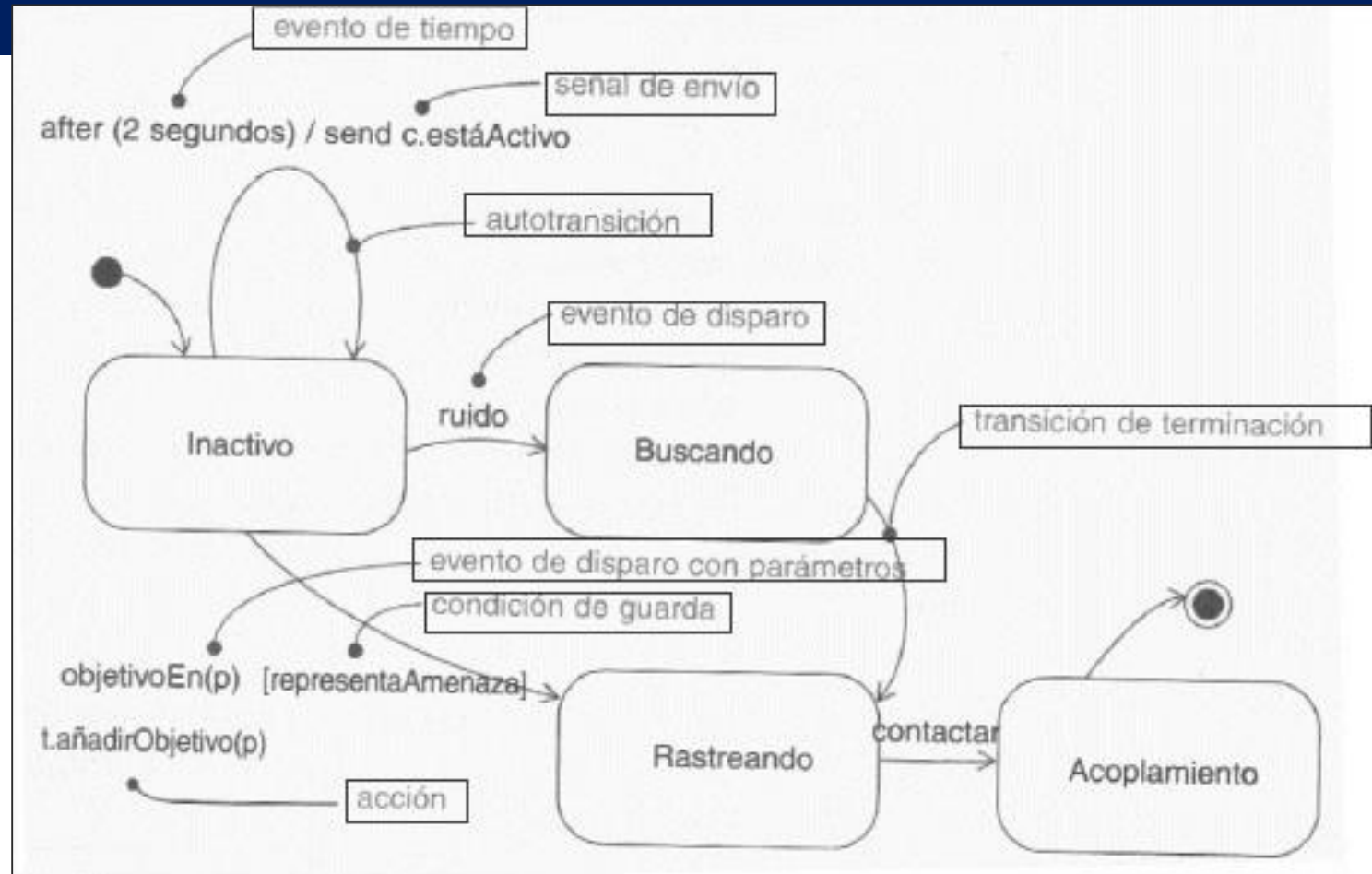


Tan pronto como finaliza la ejecución de A1, se genera un evento de finalización que inicia la transición a S2



Tan pronto como finaliza la ejecución de A1, se genera un evento de finalización; si g1 se evalúa como verdadero, se produce la transición; **Si no, esta transición nunca puede suceder.**

### 3. TRANSICIONES. EJEMPLO.



# 4. EVENTOS

Un **evento** es la especificación de un acontecimiento significativo, ubicado en el tiempo y en el espacio.

- Se utilizan en máquinas de estado para modelar la aparición de un estímulo que puede disparar la transición de un estado a otro.
- Pueden ser:

## Síncronos

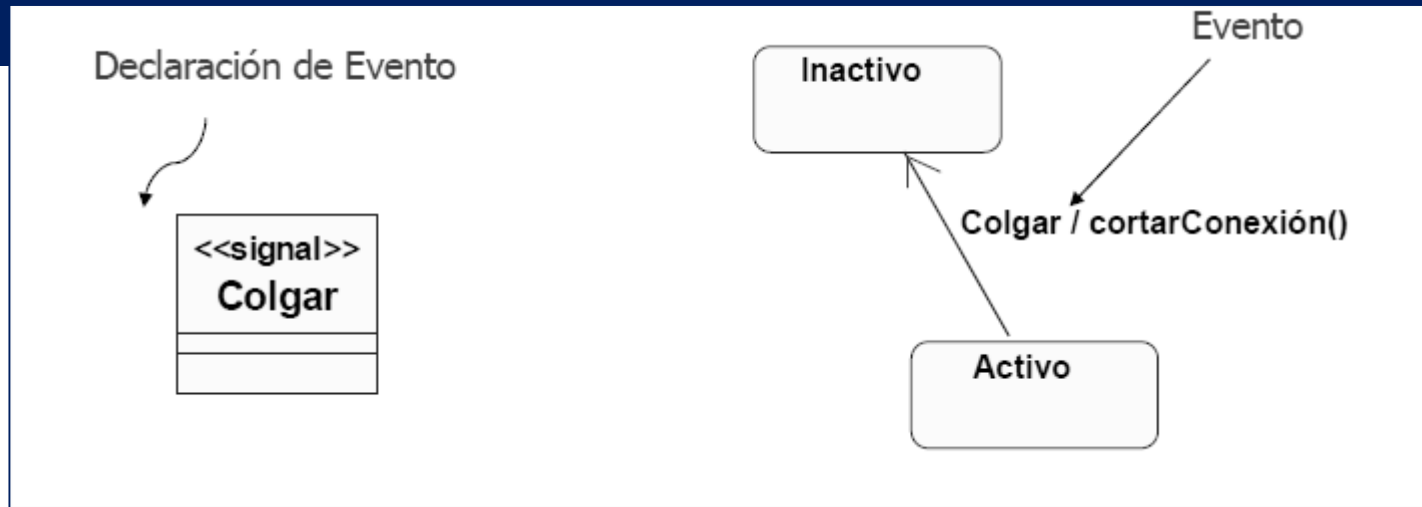
Llamadas (invocación de operaciones, ya vistos)

## Asíncronos

Señales (excepciones), Paso de tiempo, Cambio de estado.



# 4. EVENTOS



En cuanto a donde acontecen, pueden ser:

- Externos, si fluyen entre el sistema y sus actores (pulsación del ratón).

- Internos, si fluyen entre objetos del sistema (una excepción).

En UML 2 se pueden modelar cuatro clases de eventos:

- De Señal: Recepción de una comunicación asíncrona, explícita y con nombre, entre objetos.

- De Llamada: Recepción, por un objeto, de una petición explícita síncrona.

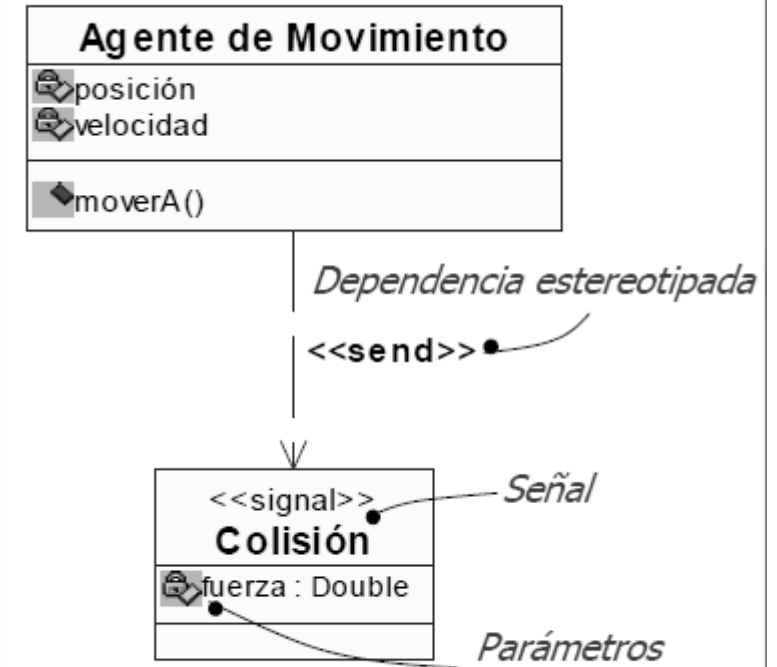
- De Tiempo: Llegada de un tiempo absoluto o transcurso de una cantidad relativa de tiempo.

- De Cambio: Un cambio en el valor de una expresión booleana.

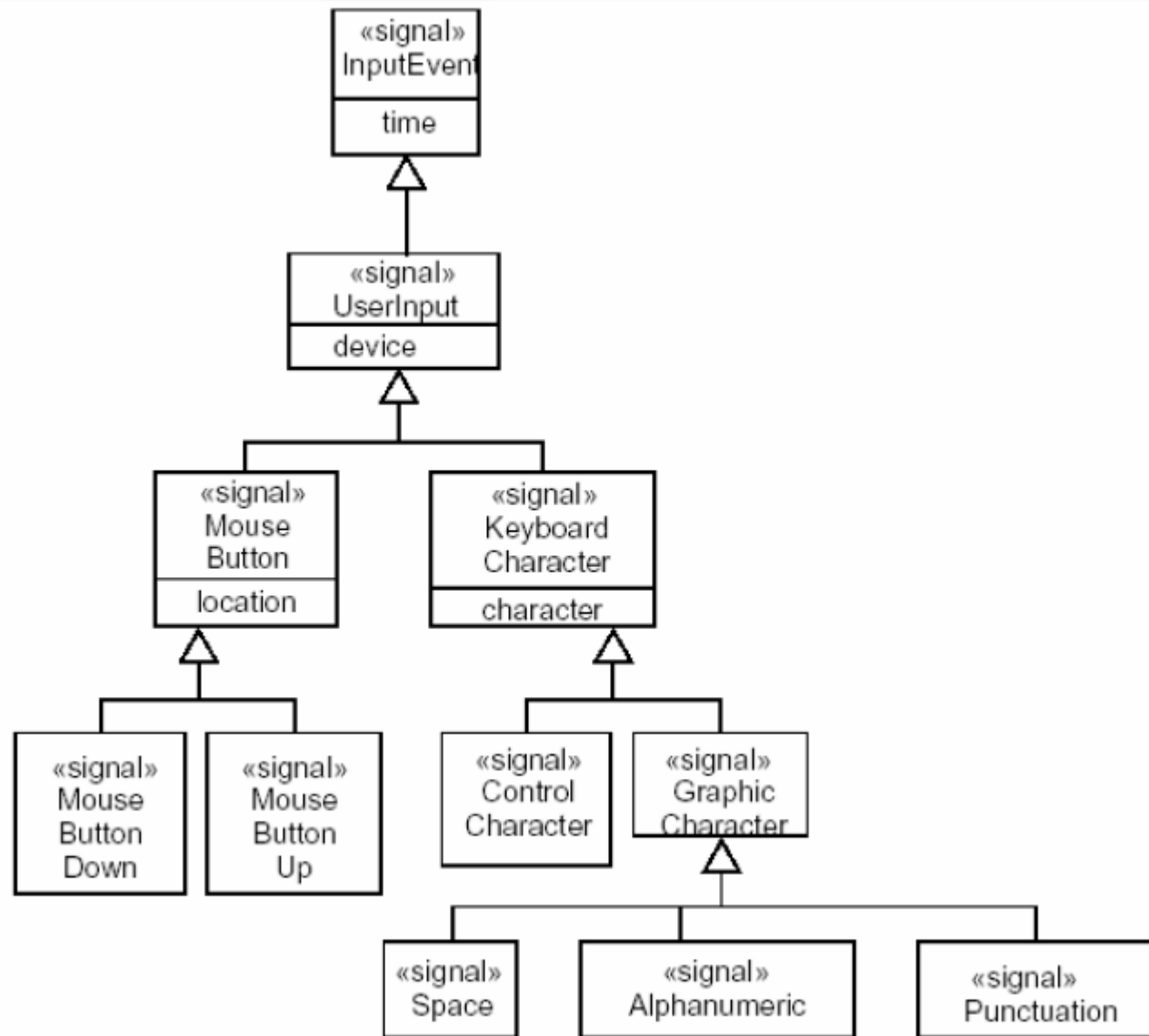
# 4. EVENTOS

## Señal

Para indicar que **una operación envía una señal** se puede utilizar una dependencia estereotipada como «send».



# 4. EVENTOS

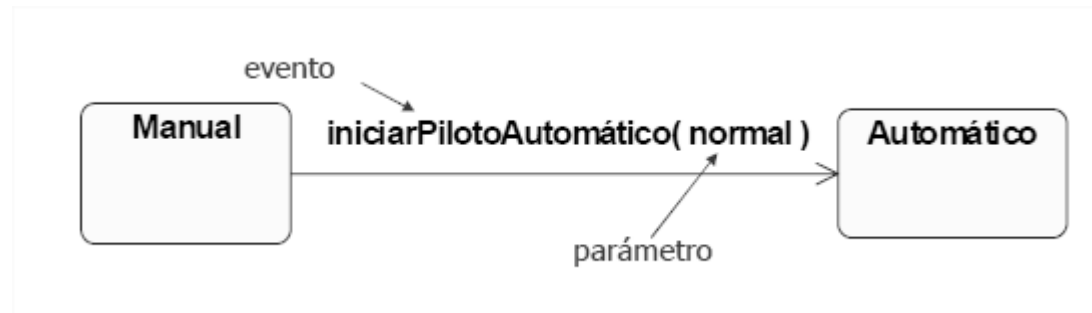


# 4. EVENTOS

## Llamada

El modelado en UML 2 de un evento de llamada es similar al de una señal.

Se muestra el evento con sus parámetros, **como el disparador** de una transición de estado.



# 4. EVENTOS

## Tiempo

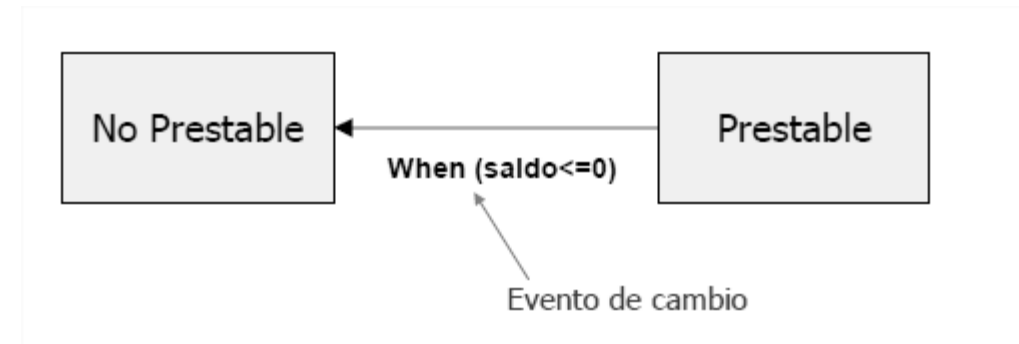
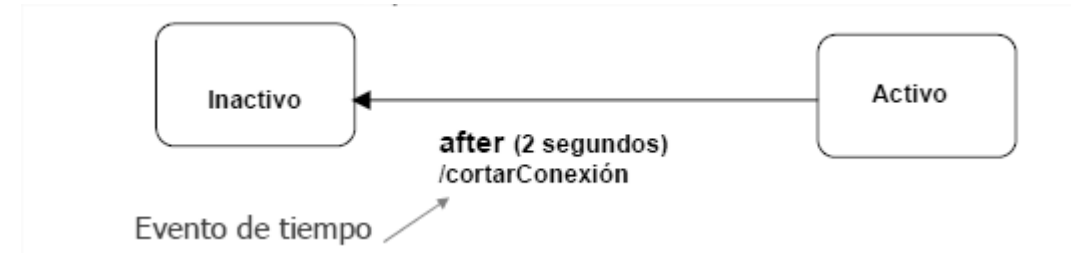
Un evento de Tiempo representa un instante en el tiempo mediante una expresión.

## Cambio

Un evento de Cambio representa un cambio de estado o el cumplimiento de alguna condición

Los eventos de **señal o llamada** implican al menos dos objetos:

Emisor (el que envía la señal o invoca la operación), y Receptor.

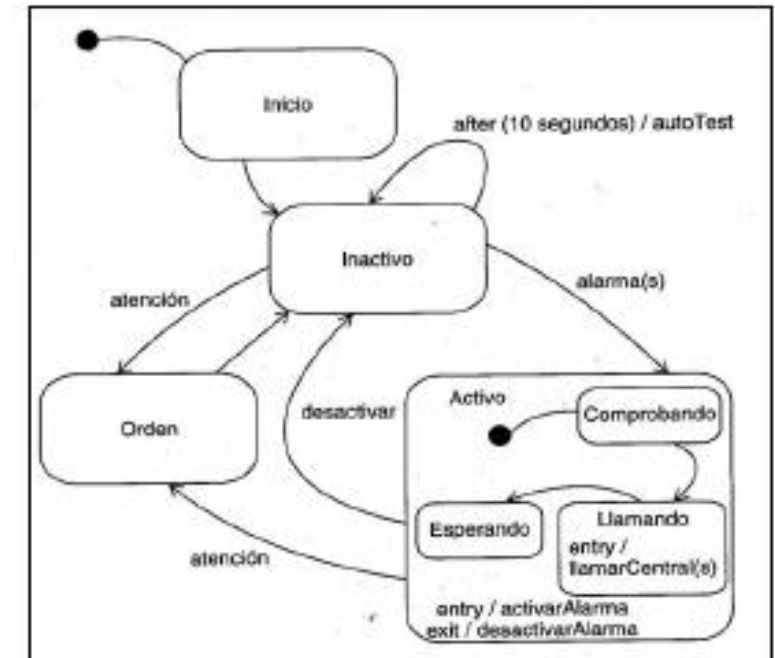


# MÁQUINAS DE ESTADO

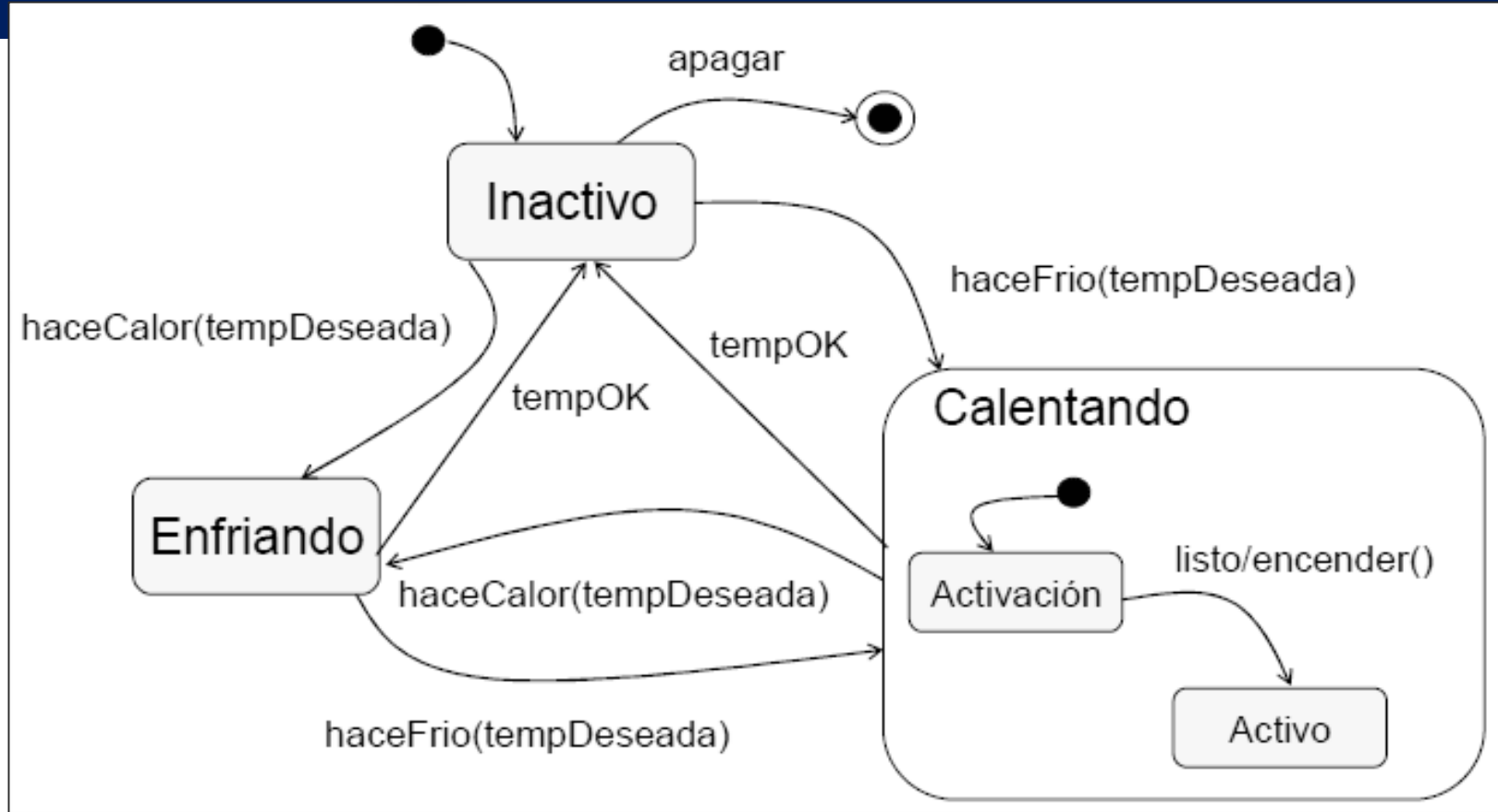
Una máquina de estados tiene forma de grafo dirigido con diferentes tipos de arcos y de nodos.

Los **nodos** representan los distintos estados por los que puede pasar un objeto.


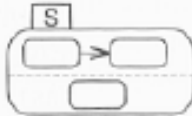
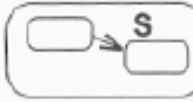

Los **arcos** se corresponden con las transiciones entre dichos estados. Normalmente, estas transiciones serán disparadas por eventos.



# MÁQUINAS DE ESTADO. EJEMPLO

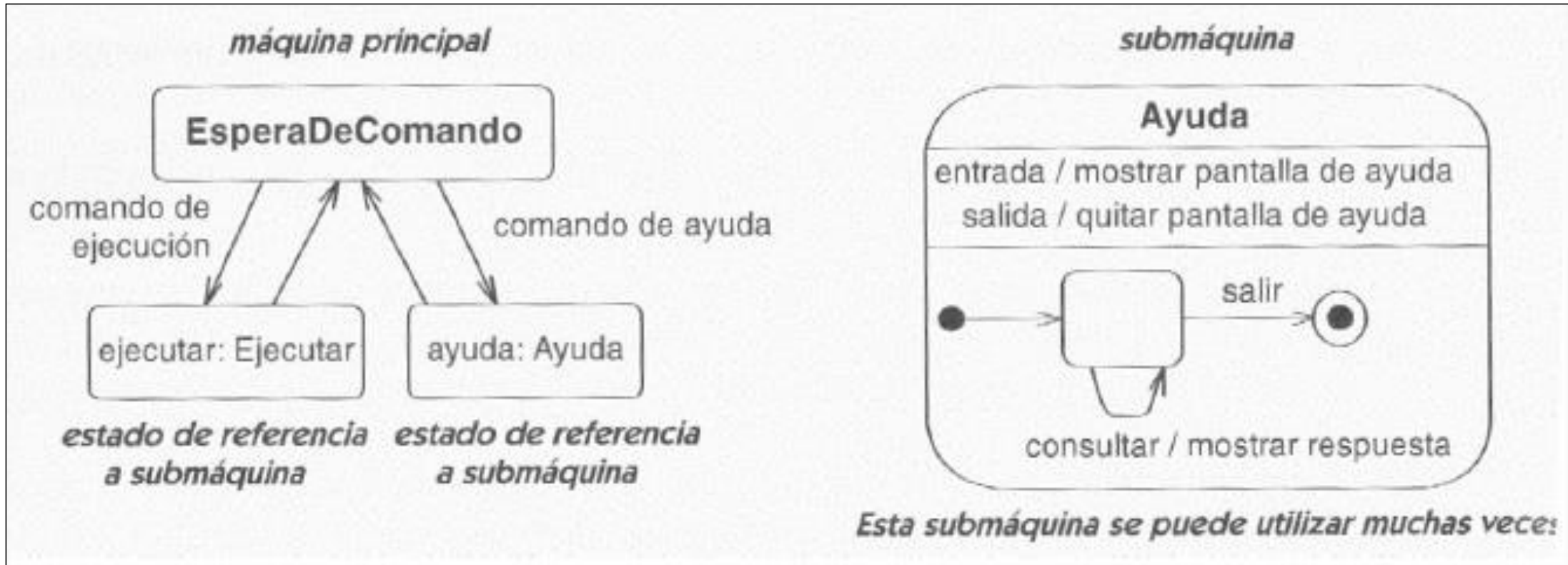


# MÁQUINAS DE ESTADO. TIPOS

Tipo de Estado	Descripción	Notación
<b>Simple</b>	Sin estructura interna.	
<b>Compuesto</b> El estado tiene estructura interna con varios estados interiores (subestados).	<b>Estado Ortogonal</b> (subestados concurrentes): Se divide en dos o más regiones. Cuando el estado está activo significa que lo está uno de los subestados de cada región.	
	<b>Estado No Ortogonal</b> (subestados secuenciales): Contiene uno o más subestados directos. Cuando el estado está activo significa que lo está uno y solo uno de los subestados.	
<b>Submáquina</b>	Semánticamente, un estado submáquina es equivalente a un estado compuesto. Se utiliza para factorizar comportamiento (cuando el mismo comportamiento debe aparecer en varios lugares). Una submáquina puede ser referenciada desde dentro de otras máquinas de estado.	



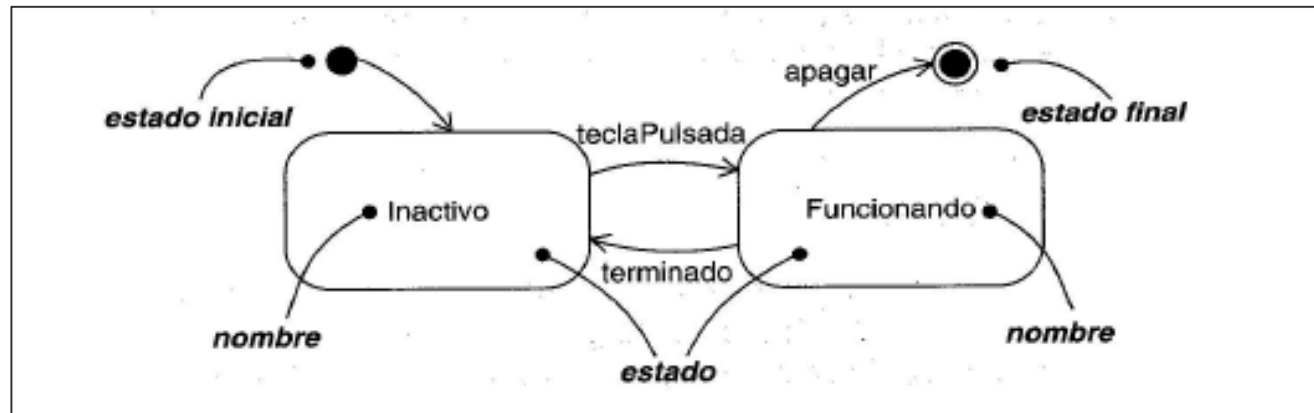
# MÁQUINAS DE ESTADO. EJEMPLOS



# MÁQUINAS DE ESTADO.

**Inicial:** Indica el punto de comienzo por defecto para la máquina de estados o para el subestado.

**Final:** Indica que la ejecución de la máquina de estados o estado que lo contiene, ha finalizado. Si la máquina tiene uso infinito, puede no tener estado final (pero siempre tendrá estado inicial).



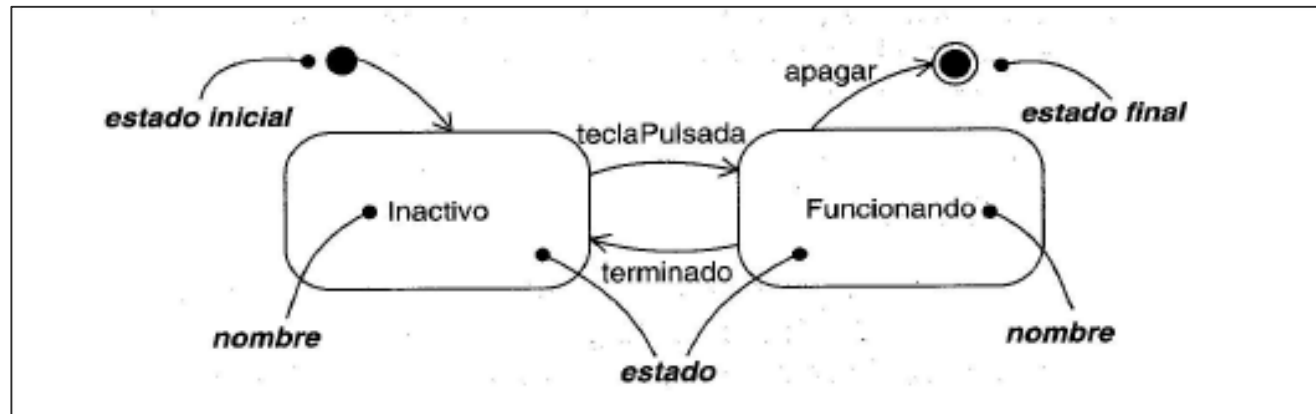
# MÁQUINAS DE ESTADO.

Un **subestado** es un estado anidado dentro de un estado compuesto.

Los subestados dentro de un estado compuesto pueden ser concurrentes (estado compuesto ortogonal) o secuenciales (estado compuesto no ortogonal).

En UML 2, un estado compuesto se representa igual que un estado simple, pero con una máquina de estados anidada.

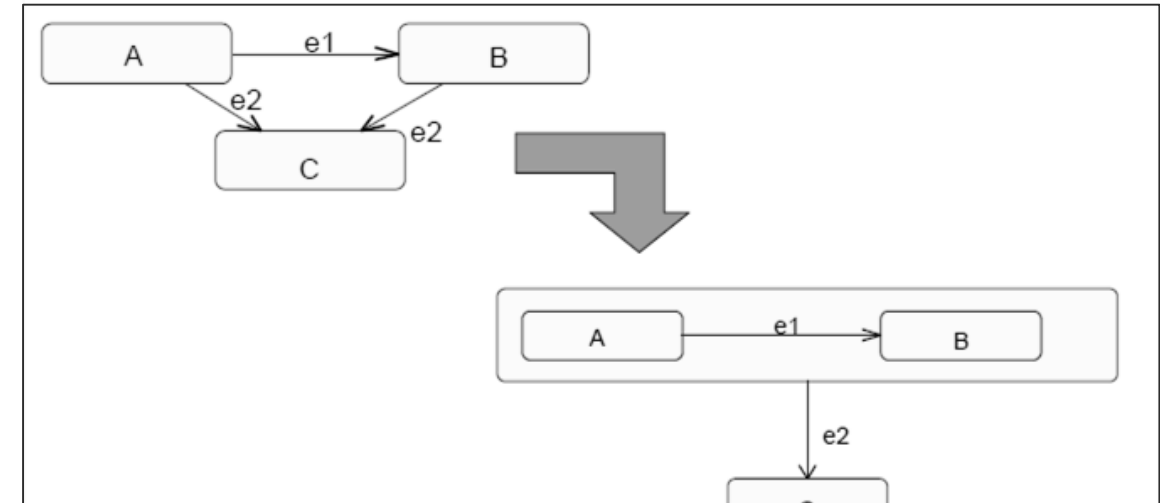
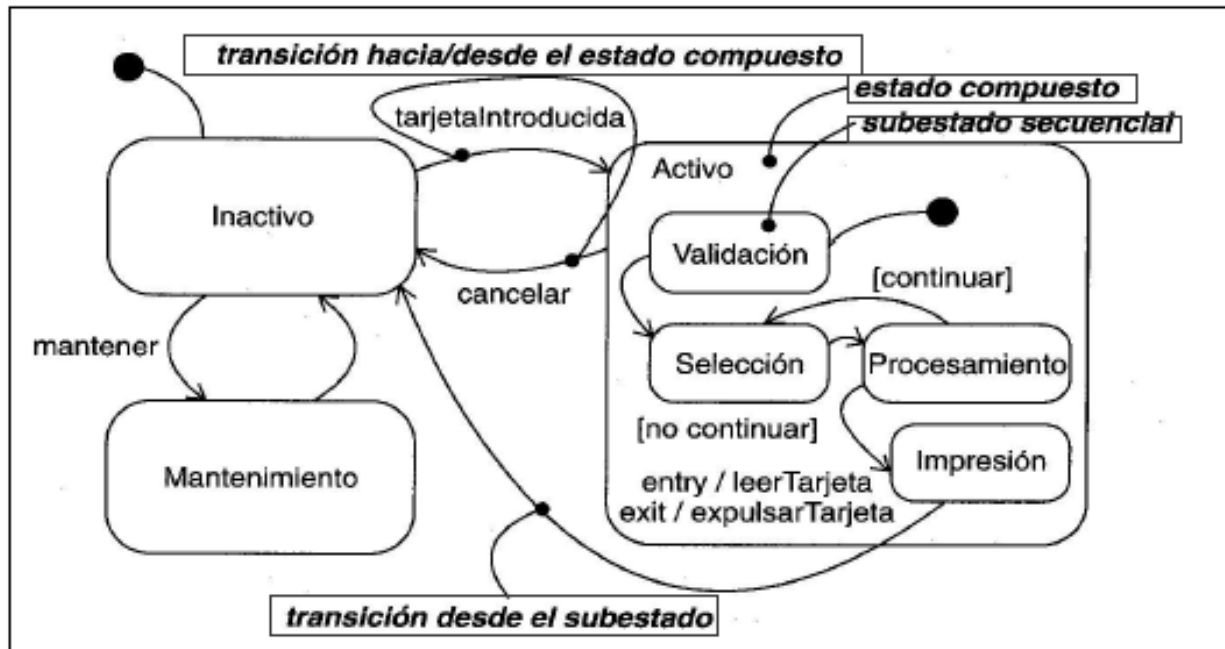
Los subestados se pueden anidar a cualquier nivel.



# MÁQUINAS DE ESTADO.

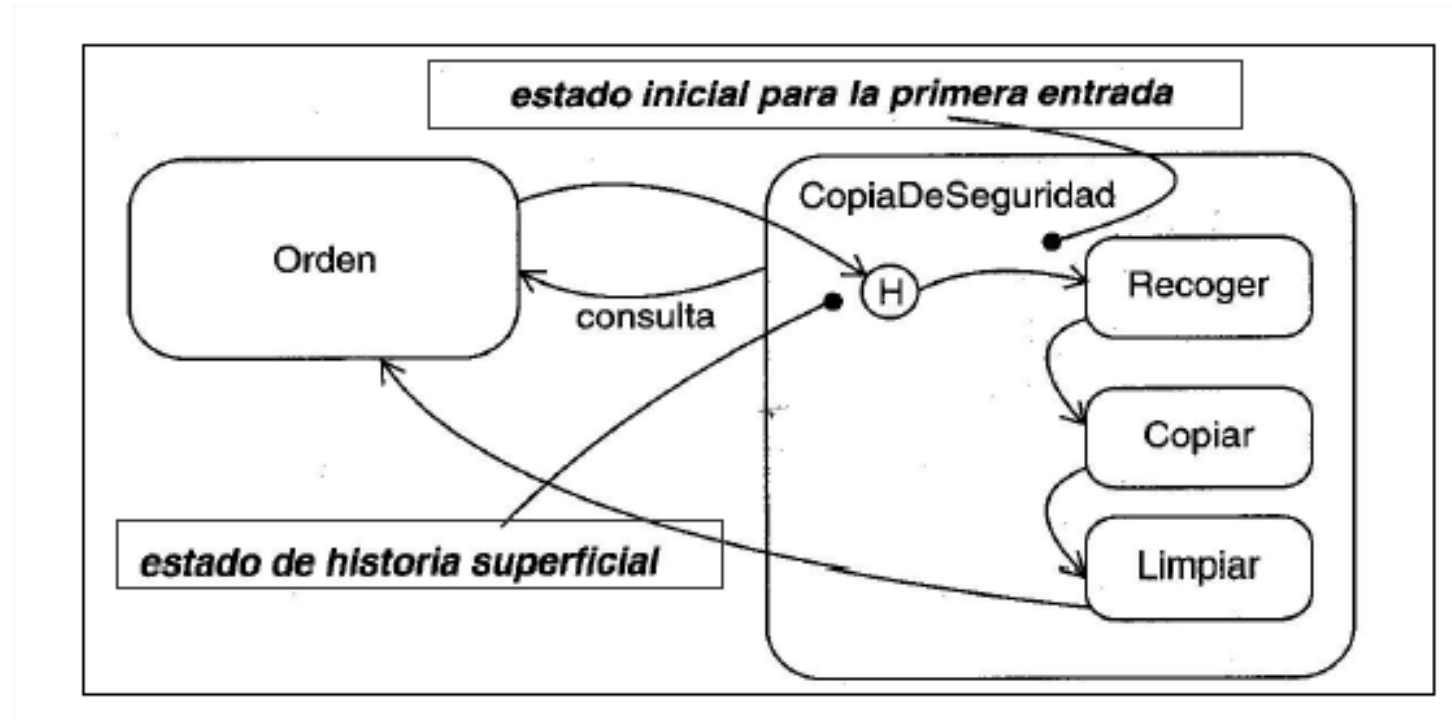
Ejemplo cajero automático.

**Subestados secuenciales.** Este tipo de estados compuestos es una ayuda para simplificar máquinas de estado mediante un mecanismo de abstracción de agregación de estados dependientes.



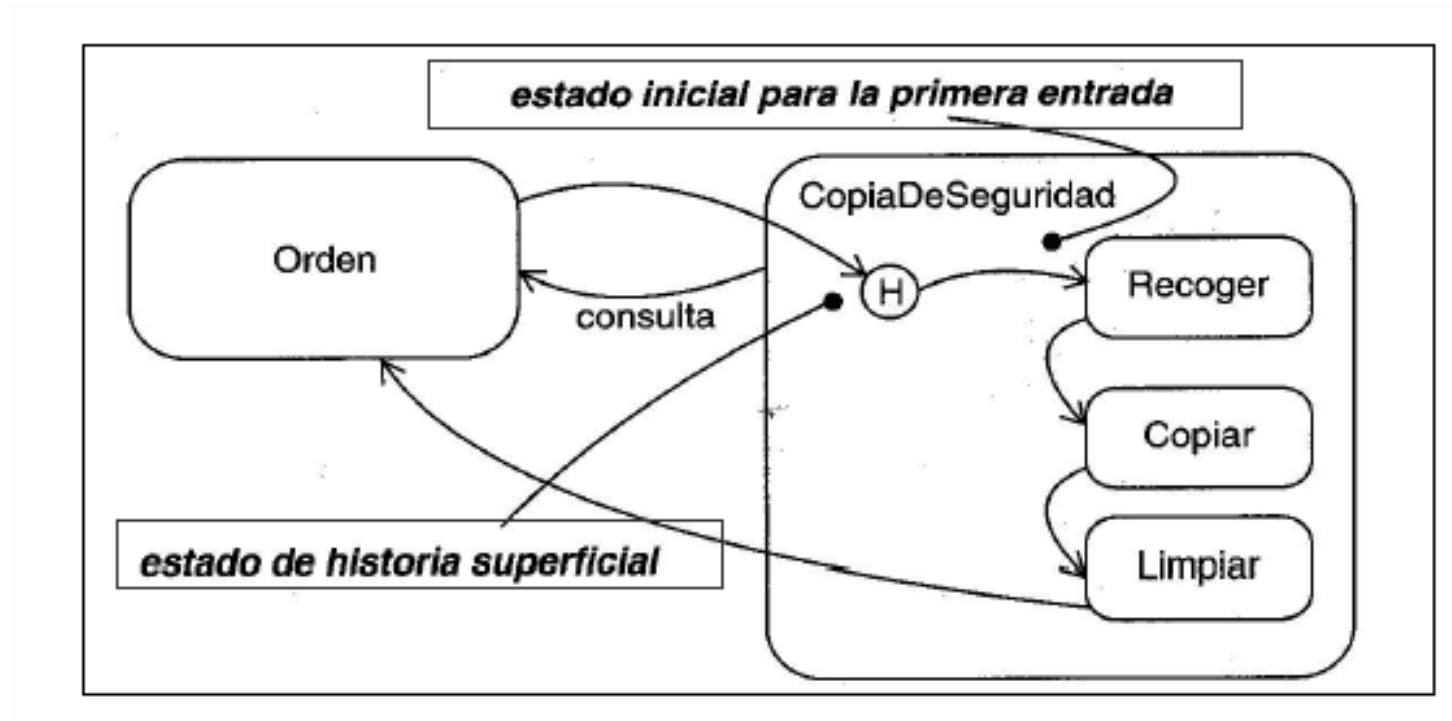
# MÁQUINAS DE ESTADO.

**Estados con historia.** Un estado compuesto recuerda el último subestado activo antes de la transición que provocó la salida del estado compuesto.



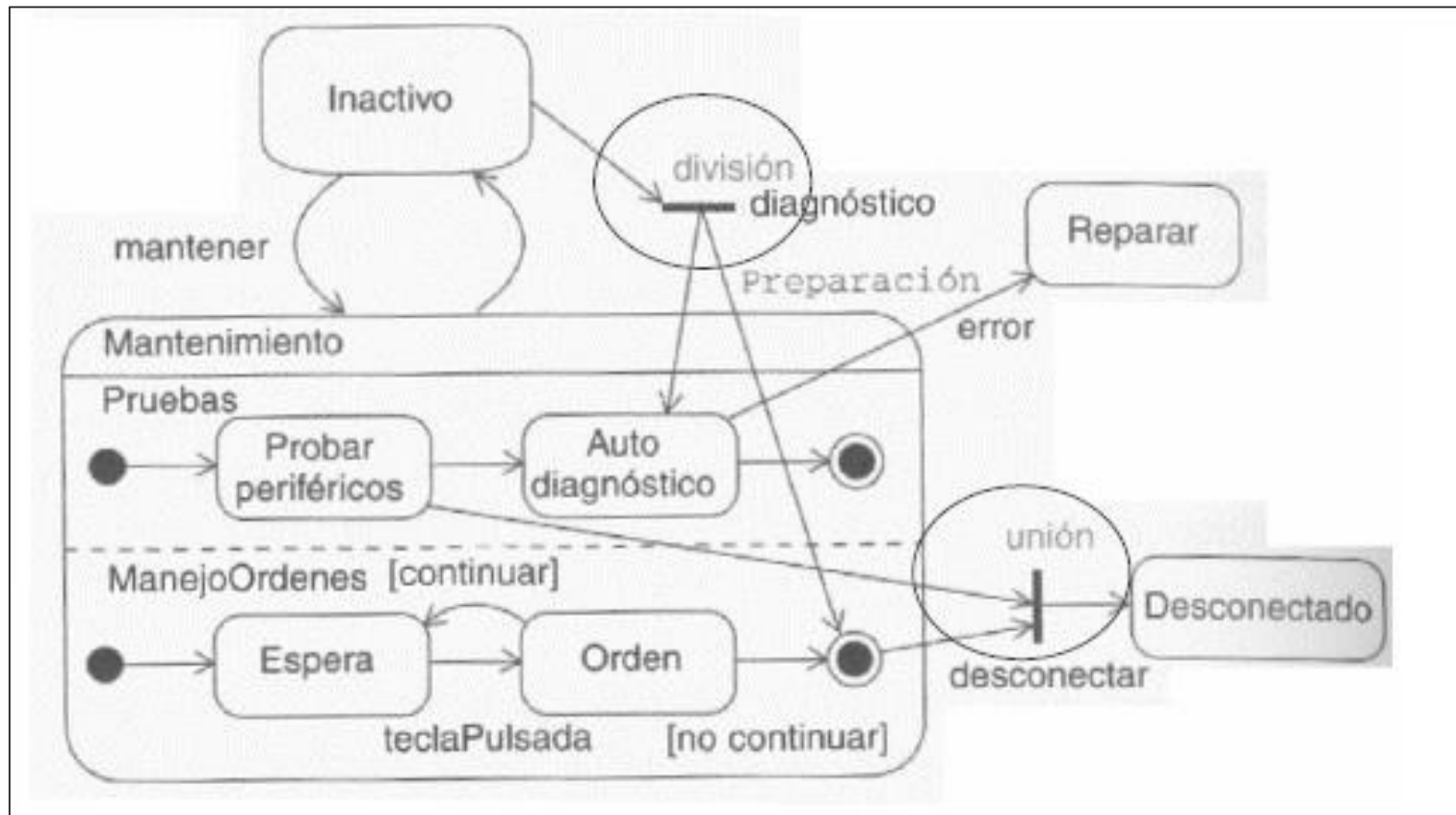
# MÁQUINAS DE ESTADO.

**Subestados Concurrentes.** Transición de **División** (el control pasa de un estado simple a varios estados) y Transición de **Unión** (varias entradas, cada una de un subestado de una región diferente, pasan el control a un único estado simple).



# MÁQUINAS DE ESTADO.

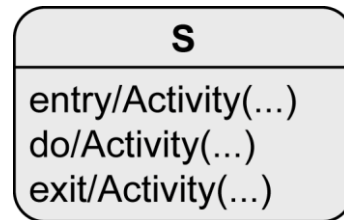
Subestados Concurrentes. Transición de **División** (el control pasa de un estado simple a varios estados) y Transición de **Unión** (varias entradas, cada una de un subestado de una región diferente, pasan el control a un único estado simple).



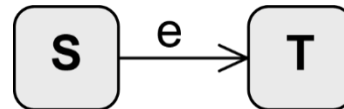
# NOTACIÓN

Name	Notation	Description
------	----------	-------------

State
-------



Transition
------------



Initial state
---------------



Final state
-------------

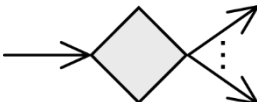
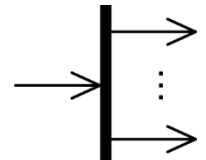
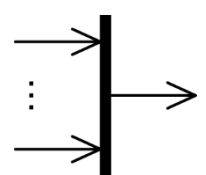
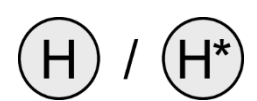


Terminate node
----------------





# NOTACIÓN

Name	Syntax	Beschreibung
Nodo decisión		Nodo desde el cual múltiples alternativas las transiciones pueden originarse
Nodo en paralelo		División de una transición en múltiples transiciones paralelas
Nodo sincronizado		Fusión de múltiples transiciones paralelas en una transición
Estado de historia superficial / profunda		"Dirección de devolución" a un subestado o un subestado anidado de un estado compuesto