Seminario 4. Diagramas de estado

Ingeniería de Requisitos – Curso 2023 / 2024

Departamento de Informática

Universidad de Oviedo

Diagramas de estado

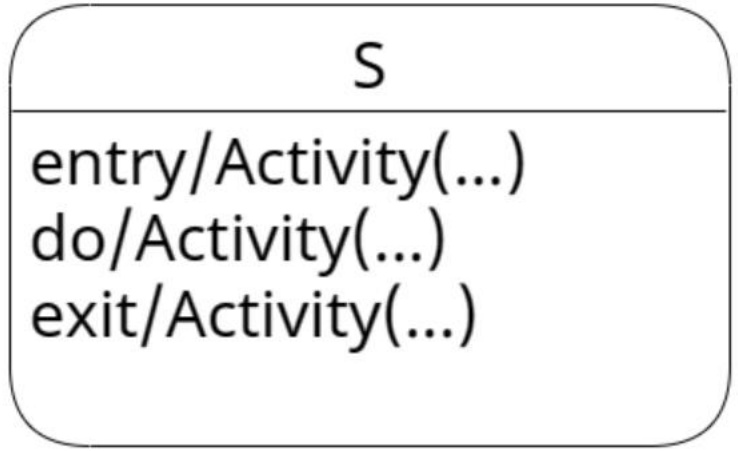
Todos los objetos de un sistema pasan por una serie de estados finitos.

Nuestro objetivo es representar esos posibles estados del sistema u objeto con un diagrama UML.

Estado

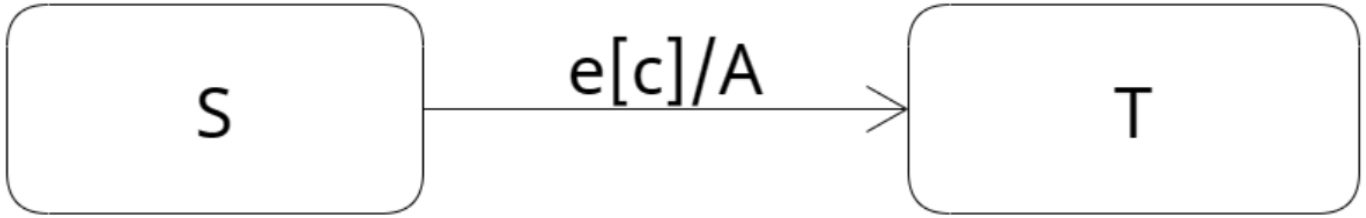
Describe el estado en el que se encuentra un objeto en un momento concreto de su ciclo de vida.

Un estado puede ejecutar actividades al entrar o salir del estado y mientras se encuentra en este.



Transición

Representa la transición entre dos estados. En la imagen se representa una transición externa.



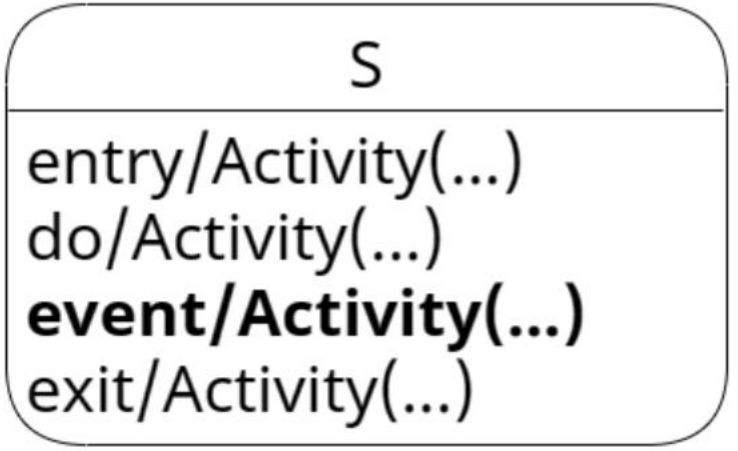
Evento (e): El evento que dispara el cambio de estado. Los eventos deben provenir de fuera del sistema u objeto que estemos modelando.

Condición (c): La condición booleana que permite el cambio de estado.

Actividades (A): Las actividades que se ejecutan durante el cambio de estado.

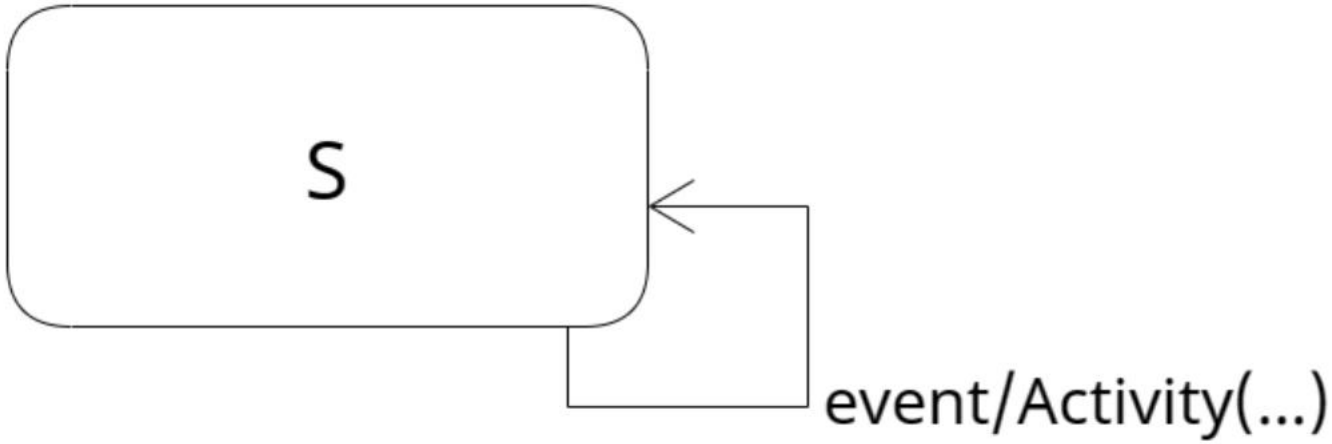
Transición interna

Se maneja la aparición de un evento dentro de un estado. Como el objeto nunca sale de dicho estado, las actividades en entrada y salida no se ejecutan.

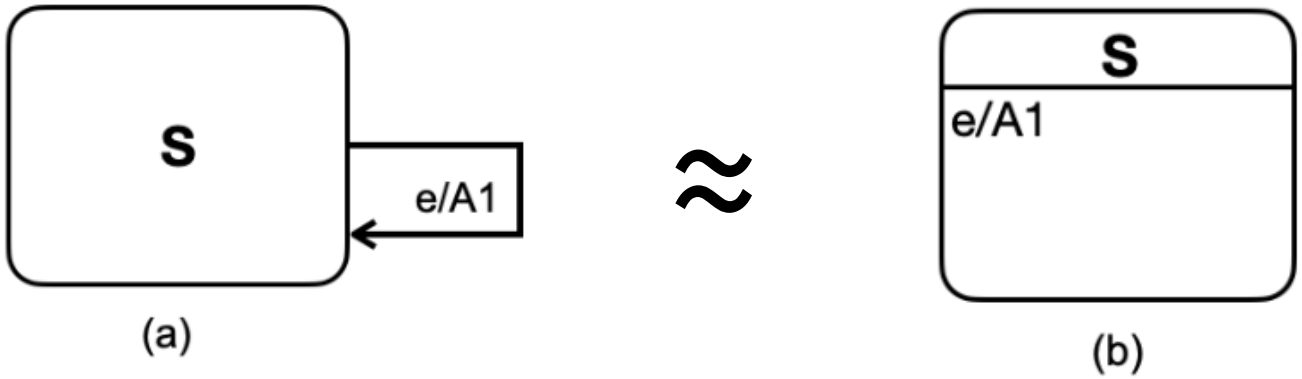


Auto-transición

Es una transición en la que el estado fuente y el objetivo son el mismo.

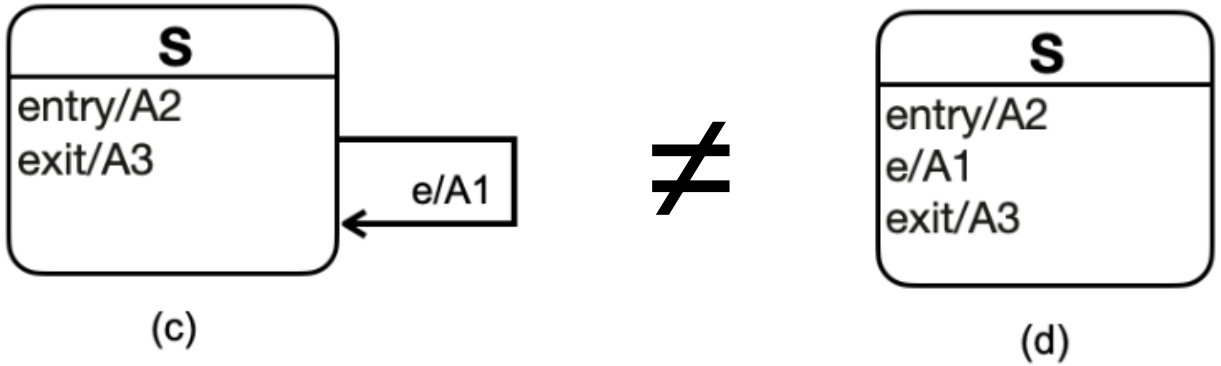


¿Son equivalentes? (1)



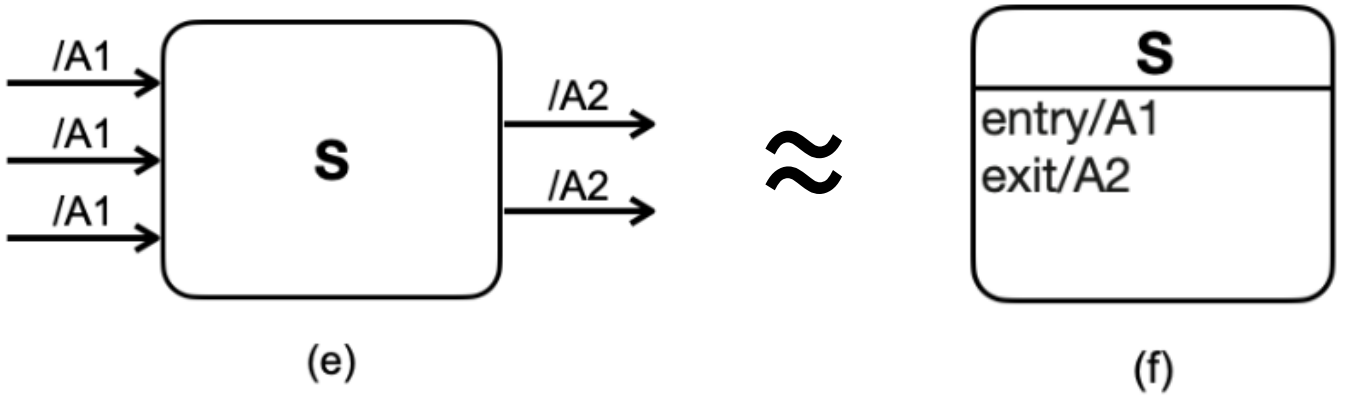
Sí, en este caso la transición interna y auto-transición son equivalentes. En ambas se ejecuta la actividad A1 cuando ocurre el evento e y el estado origen y destino son el mismo.

¿Son equivalentes? (2)



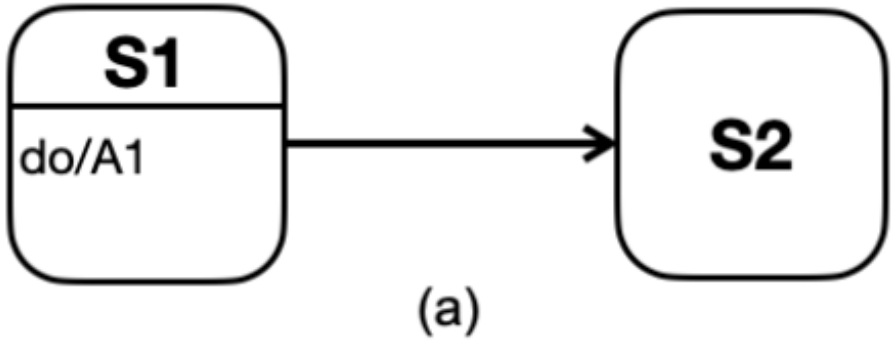
No, en la auto-transición se ejecutan las actividades de entrada y salida. Se ejecutarían A3, A1 y A2 en ese orden. En la transición interna no se activan los eventos de entrada y salida por lo que solo se ejecuta A1.

¿Son equivalentes? (3)



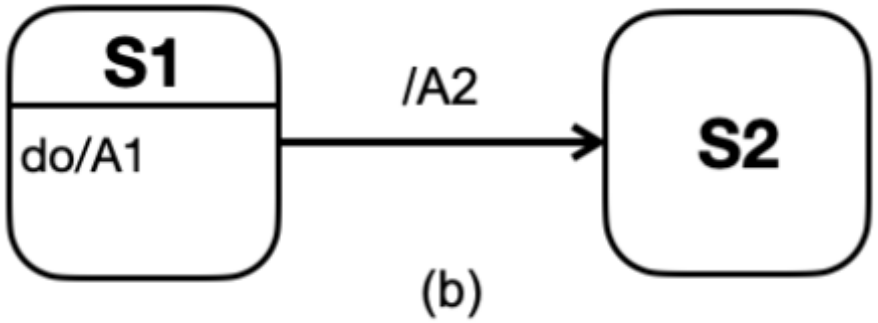
Sí, al ejecutarse la misma actividad siempre que se pasa al estado S y la misma actividad siempre que se sale de este, sería equivalente a poner estas actividades como entrada y salida del estado.

Explica el orden de ejecución (1)



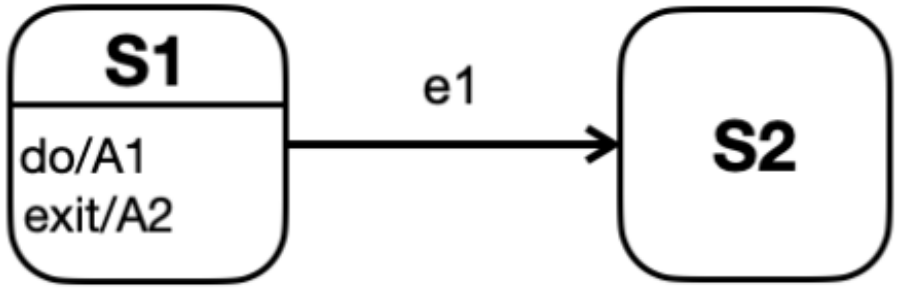
La transición ocurre en cuanto se termina la actividad 1.

Explica el orden de ejecución (2)



La transición ocurre en cuanto se termina la actividad 1 y la actividad 2 se ejecuta durante la transición.

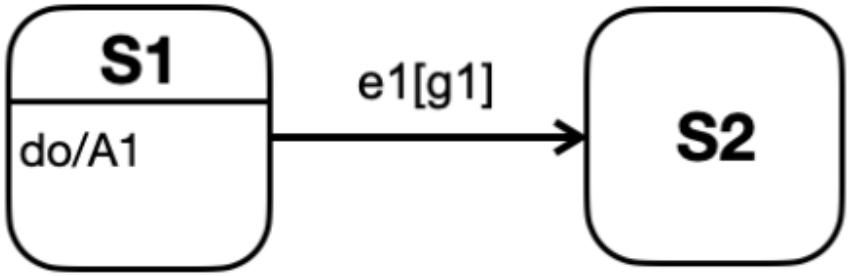
Explica el orden de ejecución (3)



(c)

La transición ocurre cuando pasa el evento. En cuanto ocurre el evento se para la ejecución de la actividad 1, y al salir del estado S1 se ejecuta la actividad 2.

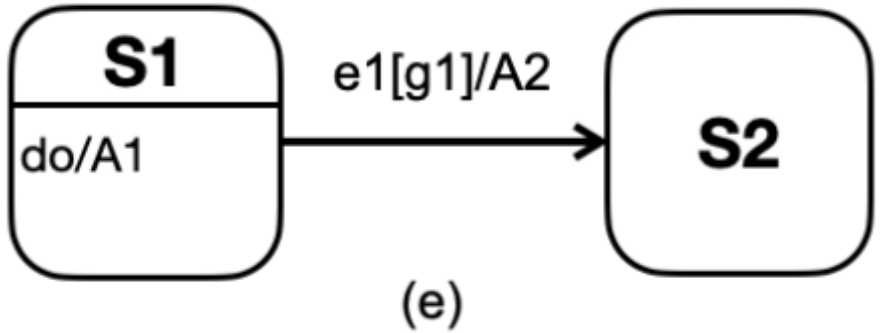
Explica el orden de ejecución (4)



(d)

La transición ocurre cuando pasa el evento e1, pero solo si se cumple la condición g1. Si se cumple, se interrumpe la actividad 1 y se pasa al estado S2. Si no se cumple, se continua con la ejecución de la actividad 1.

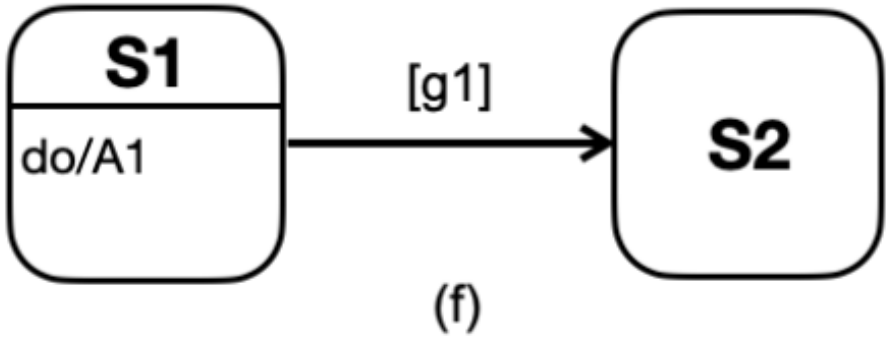
Explica el orden de ejecución (5)



(e)

Ocurre lo mismo que en el ejemplo anterior, pero si la condición g1 se cumple, se ejecuta la actividad 2 durante la transición.

Explica el orden de ejecución (6)



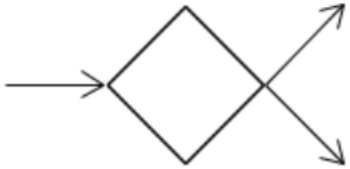
(f)

Cuando se termina de ejecutar la actividad 1, se comprueba la condición g1, si se cumple se pasa a S2, si no se cumple el objeto se queda en el S1 (esto es un problema si no tenemos otra transición).

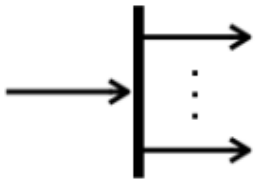
Pseudoestados

Son transitorios, el objeto no se puede quedar en uno de estos estados.

Estado inicial. Cuando el sistema esté en este nodo, pasará inmediatamente al siguiente.

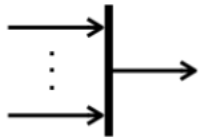


Nodo de decisión. Representa transiciones alternativas. Debe cubrir todas las alternativas posibles.



Nodo de paralelización. Divide el flujo en transiciones múltiples concurrentes. No se pueden especificar eventos ni condiciones en sus aristas.

Pseudoestados

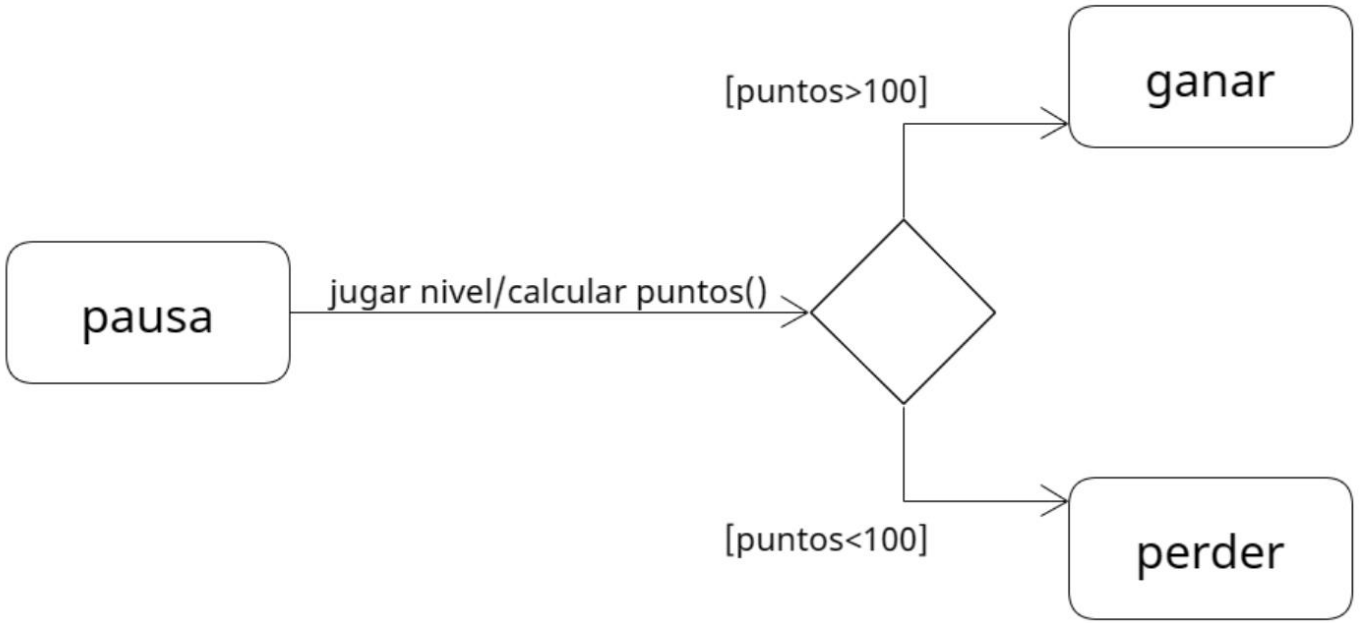


Nodo de sincronización. Une flujos concurrentes. No se pueden especificar eventos ni condiciones en sus aristas.

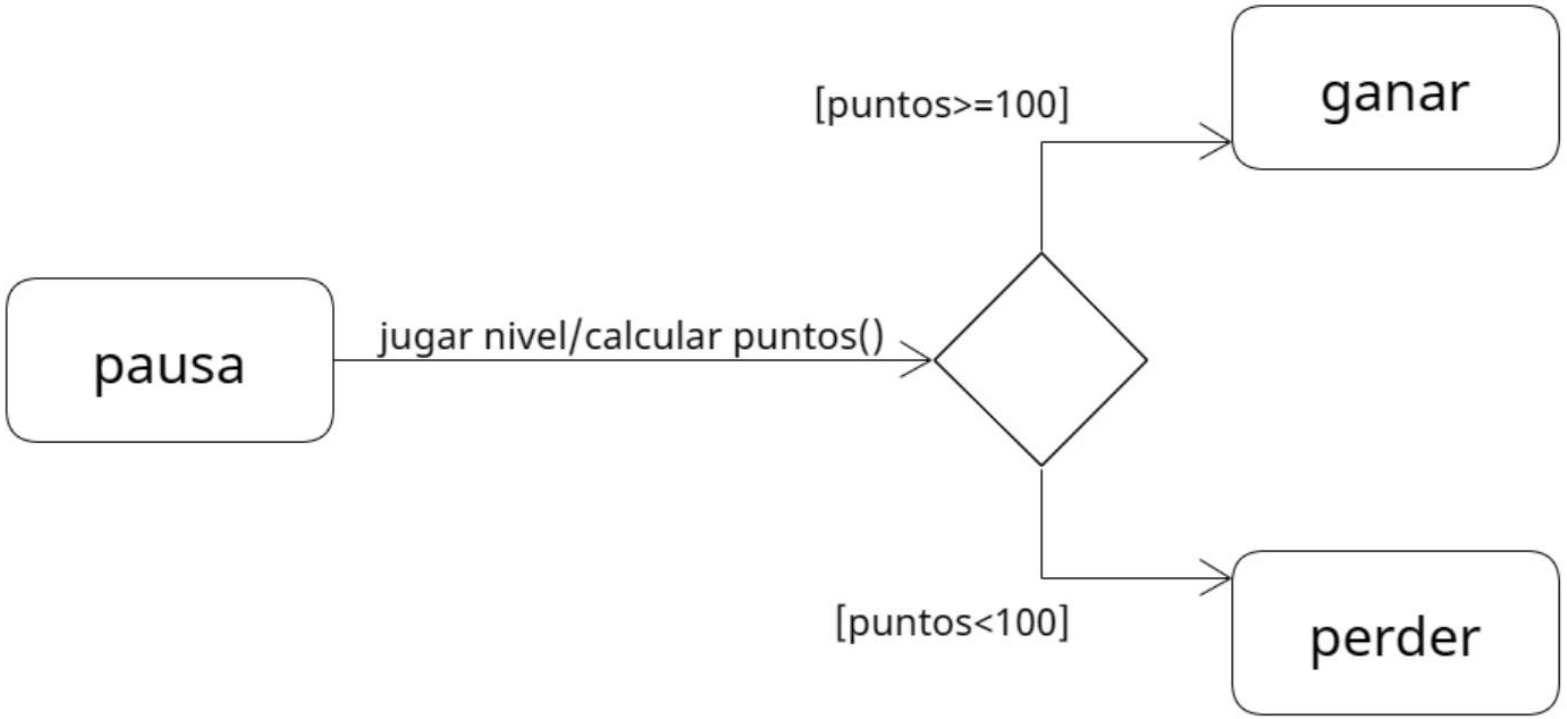
Nodo de terminación. Si en un flujo un objeto alcanza este nodo, dejará de existir y el estado máquina se termina. El objeto se elimina.

Estado final. Representa el fin de una secuencia de estados. Este NO es un pseudoestado, ya que el objeto puede permanecer en este estado de forma permanente.

¿Es correcto este fragmento?



No, no se cubren todas las alternativas posibles, y por tanto hay un caso en el que el objeto estudiante podría quedarse permanentemente en el estado sin calificar.



Se añade la condición de que los puntos puedan ser igual a 100.

Diagrama de estado: estudiante en una asignatura

Un estudiante puede matricularse en una asignatura. El estudiante puede desmatricularse siempre que haya pasado menos de un mes del inicio de las clases.

Dentro de la asignatura podrá estar calificado o no. Si su nota media es mayor o igual a 5 entonces habrá aprobado la asignatura, en caso contrario habrá suspendido.

Diagrama de estado: estudiante en una asignatura

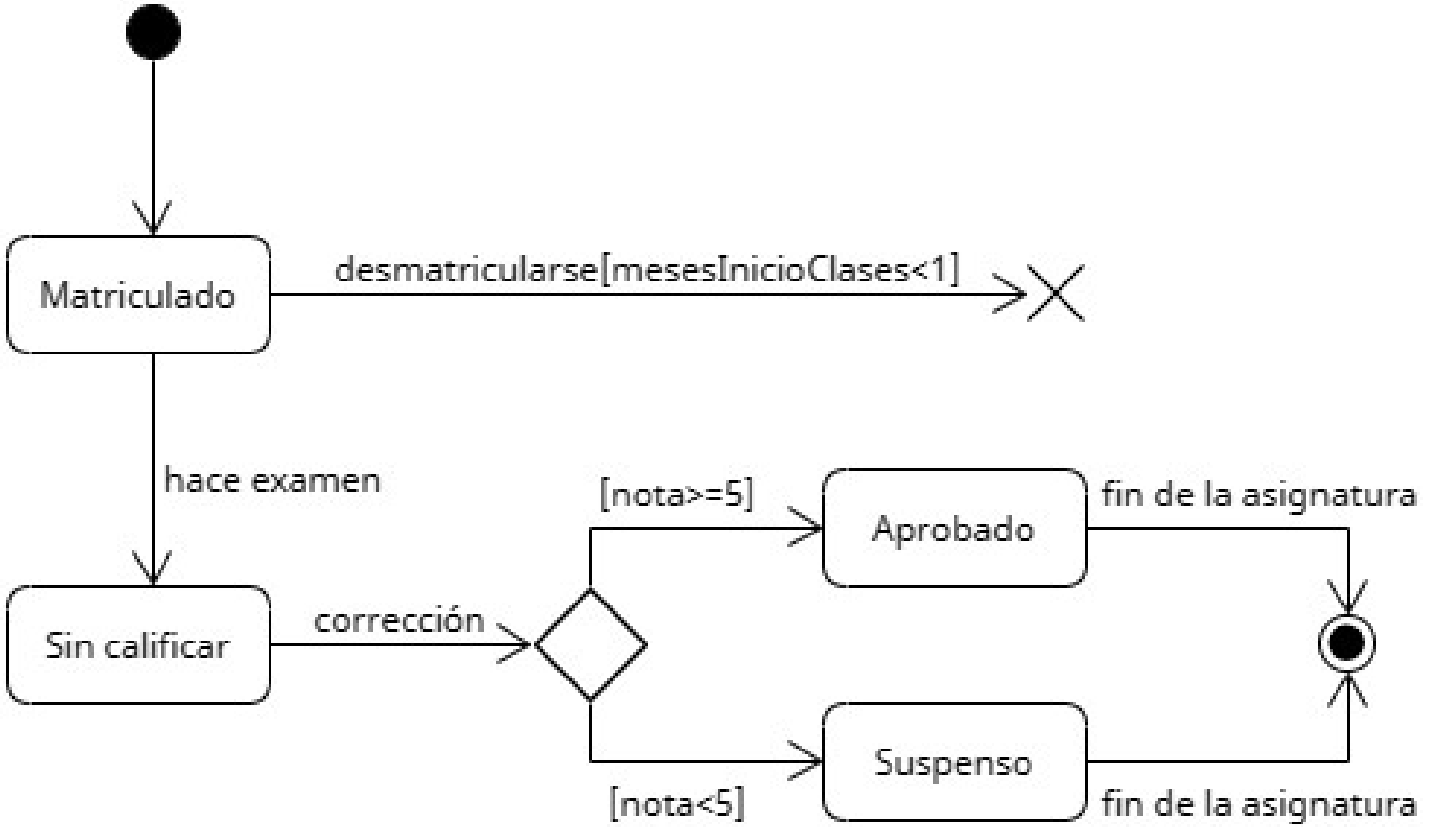
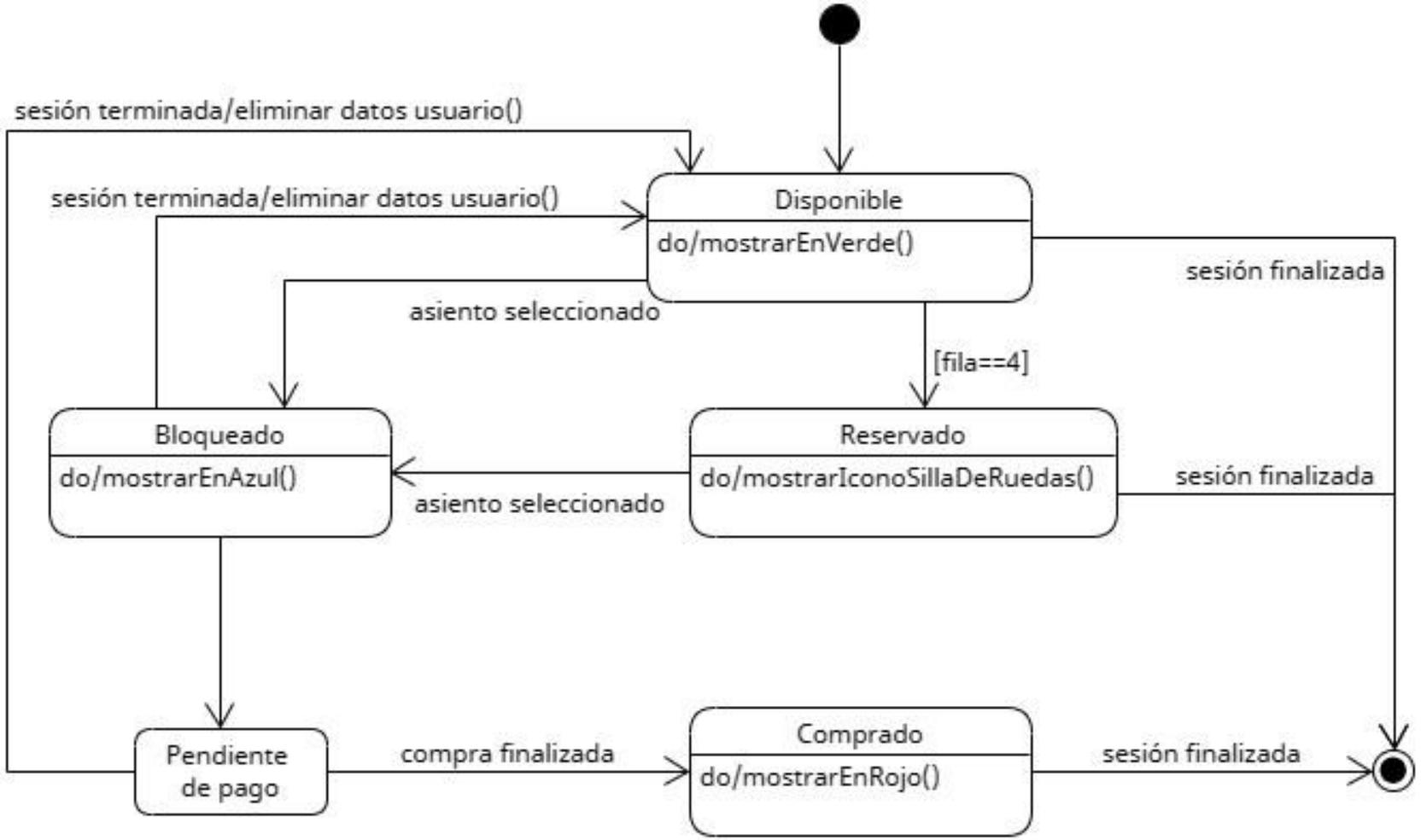


Diagrama de estado: asientos de cine

Supongamos un sistema de compra de asientos en el cine. Para cada sesión de cine habrá ciertos asientos disponibles. En todas las sesiones los asientos de la fila 4 estarán reservados para personas en silla de ruedas. Cuando los usuarios entran en la página para comprar asientos, estos se mantienen bloqueado durante el proceso de compra. La sesión caduca a los 10 minutos, tras los que volverán a estar disponibles si no se ha realizado la compra. Una vez en la fase de compra, los asientos estarán pendientes de ser pagados durante un máximo de 10 minutos. En el sistema los asientos se mostrarán de color verde si están disponibles, de color azul si están bloqueados en el proceso de compra y de color rojo si ya se han comprado. Los asientos para personas en silla de ruedas llevarán un icono con una silla de ruedas.

Diagrama de estado: asientos de cine



Bibliografía

Seidl, Martina & Scholz, Marion & Huemer, Christian. (2015). UML @ Classroom. https://doi.org/10.1007/978-3-319-12742-2