Entornos de Desarrollo
Tema 7. UML (III)

1º CFGS: Desarrollo de Aplicaciones Web
IES Severo Ochoa - Elche
2011/2012

Entornos de Desarrollo

Tema 7. UML (III)

por: Javier Martín Juan

Revisión: 903aa850175c

Última actualización: 9 de enero de 2012

Reconocimientos:

- Francisco Aldarias Raya - Plantillas LaTeX

Índice

1. Objetivos

- Realizar diagramas de estados

2. Diagramas de estados

Los diagramas de estados sirven para modelar gráficamente el comportamiento de una **máquina de estados**.

Una **máquina de estados** es un sistema que puede pasar por distintos modos de funcionamiento (estados) dependiendo de eventos que causan transiciones entre éstos.

Los eventos que pueden hacer cambiar de estado a la máquina pueden ser llamadas de otros objetos, una señal (por ejemplo la pulsación de una tecla), un cambio en el valor de alguna variable o simplemente el paso del tiempo.

Dos características fundamentales de las máquinas de estados son:

- La máquina de estados sólo puede estar en un único estado a la vez.
- Para un mismo estado y un mismo conjunto de eventos, la máquina siempre deberá comportarse igual.

Diferencias entre diagramas de actividades y de estados

En el diagrama de actividades no poníamos demasiada atención a la transición entre actividades. Simplemente se empezaba una actividad cuando terminaba la anterior. Estos diagramas son útiles para modelar algoritmos o procedimientos.

Sin embargo, el diagrama de estados enfatiza las condiciones y los estímulos que deben darse para cambiar entre un estado y otro.

Otra diferencia es que los diagramas de actividades pueden involucrar a varias objetos que se transfieren el control unos a otros, mientras que una máquina de estados suele estar implementada en un único objeto.

Elementos del diagrama

- *Estados*: Se representan con un rectángulo con los bordes redondeados. Dentro del estado escribimos un nombre descriptivo del mismo. Opcionalmente podemos dividir el rectángulo con una línea horizontal, y debajo de ésta podemos escribir *acciones* que se llevarán a cabo mientras la máquina está en ese estado. Las acciones pueden ir acompañadas de una *etiqueta*. Hay 4 etiquetas estándar en UML:
  - ⋆ *entrada*: La acción se ejecuta al entrar al estado.
  - ⋆ *salida*: La acción se ejecuta al salir del estado
  - ⋆ *haz*: La acción se ejecuta después de la *entrada* hasta que termina o algún otro evento la interrumpe.
  - ⋆ *incluye*: La acción hace referencia a otra máquina (o submáquina) con sus propios estados.

  La sintaxis para indicar una *acción* es:

```
etiqueta ( parametros ) [ condicion de guarda ] / accion
```

  Donde:

  - ⋆ *etiqueta*: es una de las 4 etiquetas anteriores (entrada, salida, haz, incluye).
  - ⋆ *parámetros*: Opcionales, son valores pasados a la acción, normalmente por el evento que ha producido la transición.
  - ⋆ *condición de guarda*: Opcional. Es una expresión booleana (verdadero o falso) que indica condiciones adicionales necesarias para que pueda llevarse a cabo la acción.
  - ⋆ *acción*: Representa el comportamiento que tendrá el objeto. Puede escribirse en lenguaje natural, pseudocódigo o código real.

- *Estados inicial y final*: Se representan igual que en el diagrama de actividades: un círculo negro indica el inicio y un círculo negro rodeado por una circunferencia indica el final.
- *Transiciones*: Las transiciones se representan por una flecha entre dos estados. La flecha llevará una etiqueta con la siguiente sintaxis:

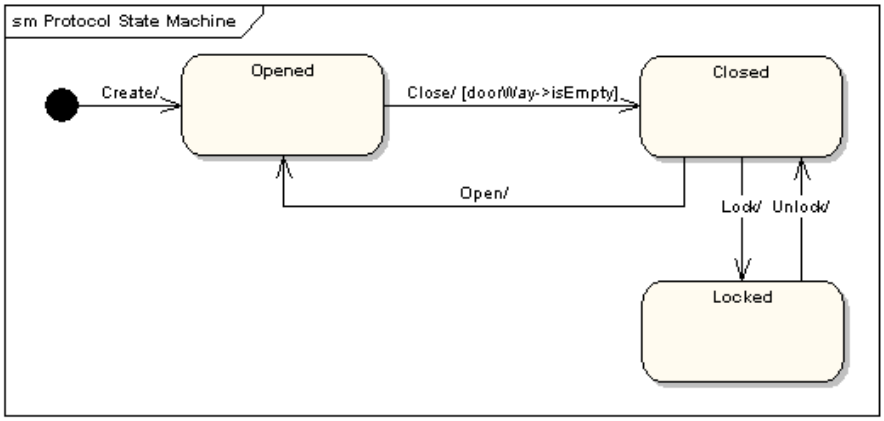
```
estimulo [ guarda ] / efecto
```

  Donde:

  - ⋆ *estímulo*: Es la condición que ha provocado la transición de estado.
  - ⋆ *guarda*: Opcional. Es una restricción que debe cumplirse para que pueda hacerse la transición. Si la condición de guarda es falsa, no se cambia de estado.
  - ⋆ *efecto*: Opcional. Indica una acción que se realizará antes de cambiar de estado.

Ejemplo: estados de una puerta

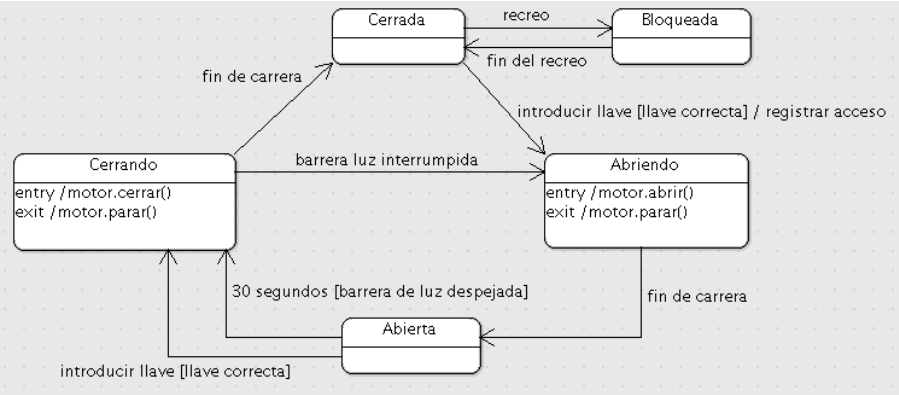
El siguiente diagrama modela los estados de una puerta a lo largo de su vida:



La puerta puede estar en 3 estados: abierta, cerrada o bloqueada. Mientras la puerta está abierta (estado "Open"), puede responder al estímulo "Close" (cerrar) y pasar al estado "Closed", siempre y cuando no haya obstáculos (condición de guarda: "doorWay isEmpty"). Desde el estado "Closed" la puerta puede responder al estímulo "Open" que volvería a ponerla en estado "Opened" o al estímulo cerrar con llave ("Lock"), que la dejaría en estado "Locked".

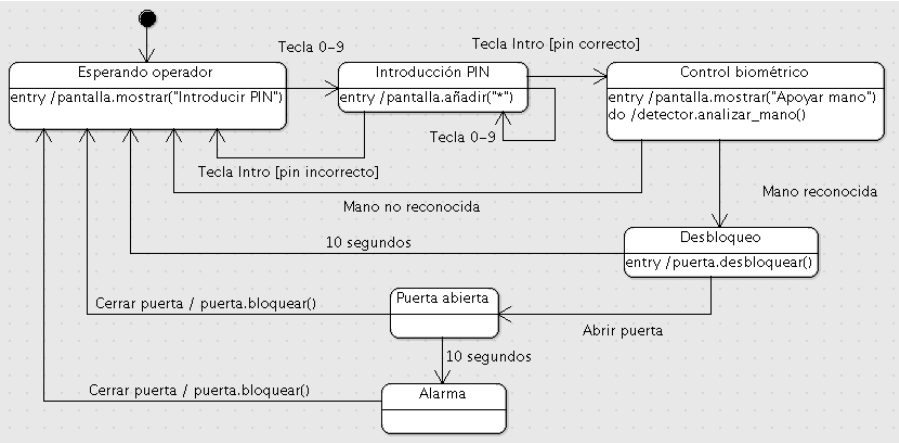
Ejemplo: puerta automática

El siguiente diagrama modela la puerta de entrada de coches del instituto. La puerta tiene un sistema de apertura por llave, un motor que mueve la puerta, dos detectores de final de carrera en los extremos del recorrido, y una pareja de emisor y receptor fotoeléctricos (barrera de luz) que reabren la puerta si alguien pasa por enmedio mientras se está cerrando. La puerta permanece abierta 30 segundos antes de cerrarse automáticamente, aunque también se puede cerrar volviendo a introducir la llave. Durante los recreos, la puerta pemanece bloqueada.



Ejemplo: control de acceso

Para entrar en un centro de datos hay una puerta que cuenta con dos sistemas de seguridad, uno biométrico que consiste en el reconocimiento de la palma de la mano, y un teclado donde el operador debe introducir su PIN. Inicialmente la pantalla indica que se debe introducir el PIN, seguido de la tecla "Intro". Si el PIN es válido, la pantalla pedirá apoyar la mano en el detector. Si la verificación es correcta, se abrirá la cerradura y se encenderá una luz verde. De lo contrario la luz será roja. Cuando el usuario cierre la puerta, ésta se volverá a bloquear. También se bloqueará si el usuario no abre la puerta en 10 segundos. Si el usuario abre la puerta pero no la cierra en 10 segundos, sonará una alarma.



3. Ejercicios propuestos

Ejercicio 7.1

Modelar un teléfono de marcación por tonos. El teléfono se puede hallar en los siguientes modos:

- Colgado
- Marcación parcial
- En conversación

Inicialmente el teléfono está colgado. Cuando se aprietan teclas, el teléfono se encuentra en modo de marcación parcial y la pantalla va mostrando los números marcados. Al apretar el botón "Llamar" el teléfono pasa a modo conversación, y la pantalla muestra la duración de la llamada. Para colgar apretamos la tecla "Colgar".

Ejercicio 7.2

Queremos modelar la pantalla del salpicadero de un coche. Dicha pantalla puede estar en tres modos: cuentakilómetros, distancia y litros/100 Km consumidos. Pulsando un botón podemos ir alternando entre los 3 modos. En el modo "distancia" podemos hacer una pulsación larga para reiniciar los contadores de distancia y los litros/100.

Ejercicio 7.3

Modelar un reloj de pulsera. El reloj tiene modo "hora", en el cual se muestra la hora, minutos y segundos. También tiene un modo "cronómetro". El cronómetro lo podemos encender, parar, y poner a cero. El tercer modo es "alarma", y en él podemos fijar una hora (HH:MM) y activar o desactivar la alarma. Por último el reloj tendrá un modo "ajuste" en el que podremos ajustar la hora y minutos. El interfaz del reloj consta de una pantalla y 3 botones.

4. Bibliografía y documentación adicional

- UML Reference Manual, 2nd Edition
  Booch, Jacobson, Rumbaugh, Ed. Addison-Wesley
- UML for Java Programmers
  Robert C. Martin, Ed. Prentice Hall
- Learning UML 2.0
  Miles, Hamilton, Ed. O'Reilly
- UML Distilled: A Brief Guide to the Standard Object Modeling Language (3rd Edition)
  Martin Fowler, Ed. Addison-Wesley
- Aprendiendo UML en 24 horas
  Joseph Schmuller, Ed. Prentice Hall
- UML 2.0 - Pocket Reference
  Dan Pilone, Ed. O'Reilly
- Análisis y Diseño Estructurado y Orientado a Objetos de Sistemas Informáticos
  Amescua et al, Ed. McGraw-Hill
- Diseño Orientado a Objetos con UML
  Raúl Alarcón, Grupo Eidos