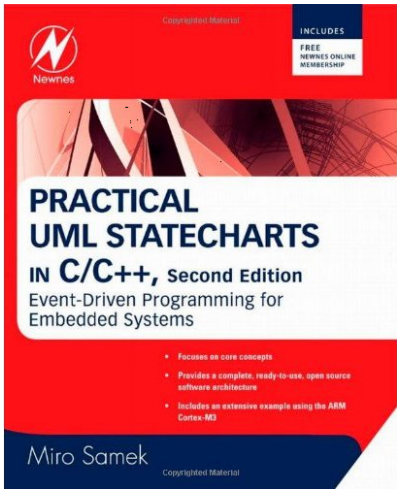
STATECHARTS
Conceptos básicos y ejemplos de uso

Statecharts

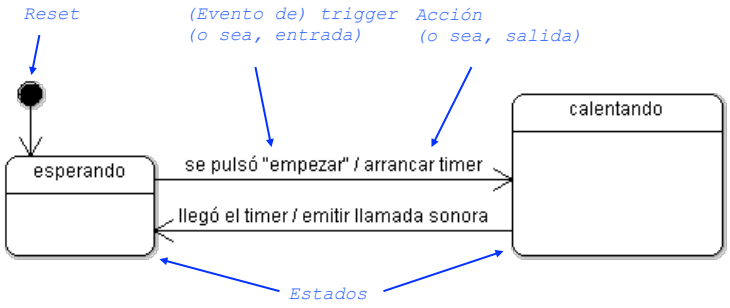
- Los diagramas de estado son particularmente útiles para modelar sistemas embebidos porque estos suelen ser reactivos
  - O sea, responden a eventos externos que no necesariamente tienen orden o periodicidad
- En 1987 Harel propone *Statecharts*, una evolución del lenguaje clásico de los diagramas de estados
  - D. Harel, "Statecharts: A visual formalism for complex systems", *Science of Computer Programming*, vol. 8 num. 3, pp. 231–274, Junio de 1987
- Para modelar sistemas medianamente complejos, los Statecharts son más compactos y claros que los diagramas de estados clásicos
  - Principalmente porque incluyen **jerarquías** y **concurrencia**
- Forman parte del UML y se usan extensivamente en herramientas MDD para embebidos

Bibliografía recomendada



- M. Samek, *Practical UML Statecharts in C/C++*, Capítulo 2
  - También está disponible como artículo en línea de EE Times
    - Ver en nuestra página de material de estudio

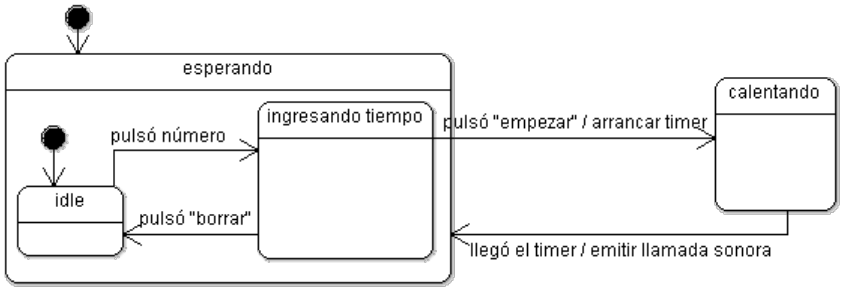
Estados y transiciones



- Como el diagrama de estados de una máquina de Mealy, pero usa rectángulos con esquinas redondeadas para los estados y un círculo lleno para el reset

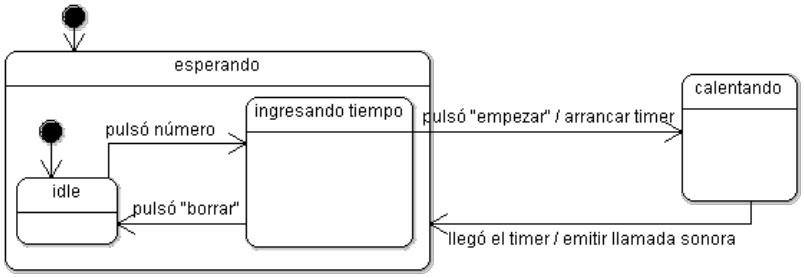
Horno microondas: Jerarquías

- Modelado de un horno microondas



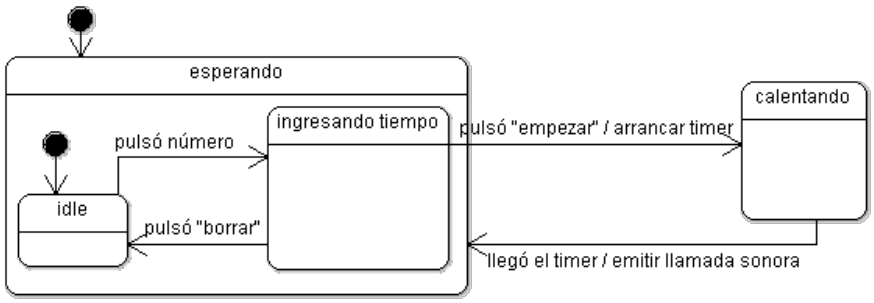
- Este tipo de jerarquización permite, en muchos casos, economizar transiciones, entre otras simplificaciones
  - que clarifican el diagrama y reducen la cantidad de errores en su edición

Horno microondas: Jerarquías



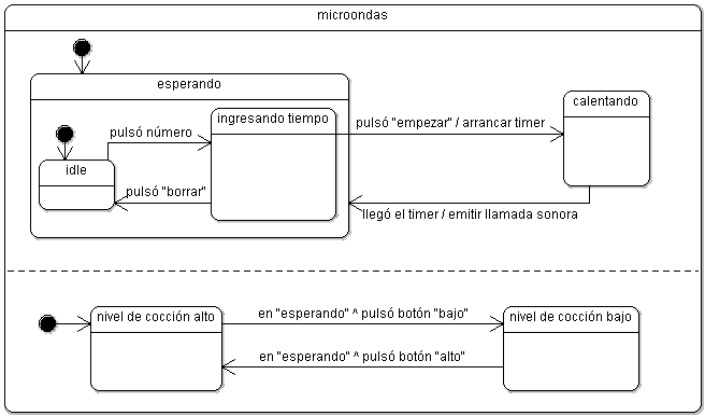
- Notar que, en el ejemplo, la transición que dispara con “pulsó ‘empezar’” sale de “ingresando tiempo”
  - O sea que si el sistema está en “idle”, pulsar empezar **no** cambia el estado
    - Al menos en lo que respecta al alcance de este diagrama
- El comportamiento en “ingresando tiempo” probablemente también sea útil modelarlo con un diagrama de estados
  - Metodología top-down

Horno microondas: Mejora



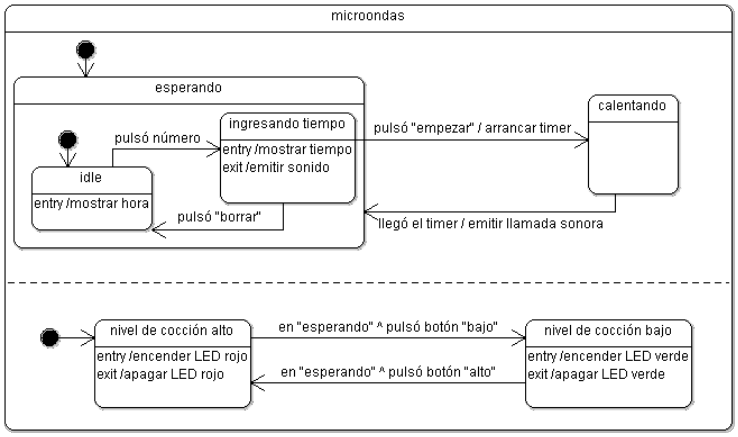
- Mejora: si no se introduce un tiempo y se presiona "empezar", arranca con tiempo = 30s. ¿Cómo podemos modificar el diagrama de arriba para incorporar ese comportamiento?
  - Puede suponerse que existe una variable "tiempo" que es modificada en "ingresando tiempo" y tomada por el timer cuando este arranca

Horno microondas: Concurrencia

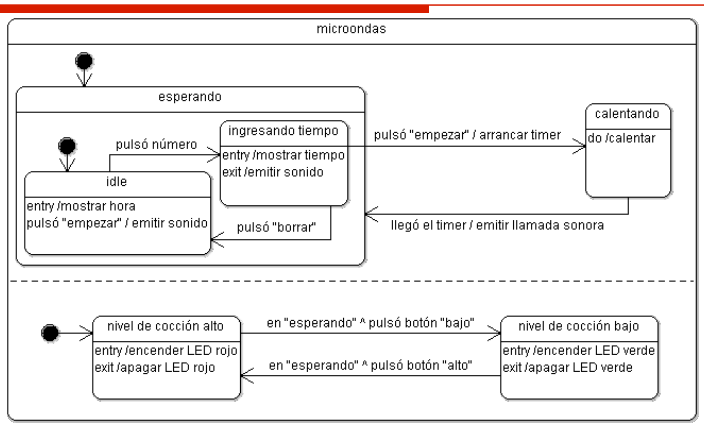


- Las dos sub-máquinas son independientes
  - Salvo la cláusula "en 'esperando'" en los triggers de abajo
- Por eso, a la concurrencia comúnmente se le dice *ortogonalidad*

Horno microondas: "Entry" y "Exit"



Horno microondas: Transiciones internas



- Notar que son similares a transiciones que empiezan y terminan en el mismo estado, salvo que no se ejecutan las acciones "entry" y "exit"
- A "calentar" normalmente se le dice actividad porque es una acción constante

Horno microondas: transiciones locales y externas

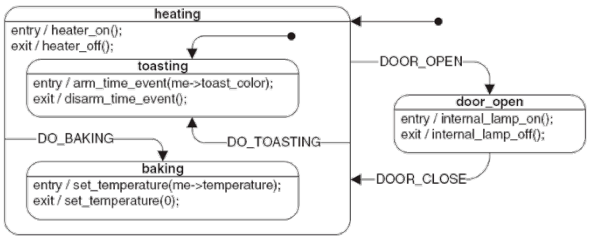
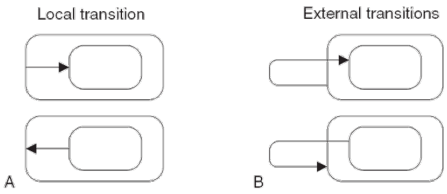


Figure 2.7: Toaster oven state machine with entry and exit actions.

- En las externas se ejecutan las acciones de "entry" y "exit", en las locales no



Pseudoestados

- Son lugares que parecen estados pero no lo son, porque el sistema no puede "estar" en ellos
  - Sirven para simplificar el diagrama e incorporarle algunas funciones
- Tipos:
  - **Inicial** (reset)
    - Se indican con un círculo lleno negro
  - **Condición** (o *choice* u opción)
    - Se indican con un diamante, un circulo vacío o uno que contiene una "C"
    - Lo usamos en la Diapositiva 6
  - **Junturas**
    - Se indican con un punto negro
    - Sirven para juntar o bifurcar transiciones cuando tienen partes en común
  - **Historia**
    - Se indican con un círculo que contiene una "H"
    - Si se entra en una transición que termina en este pseudoestado, el próximo estado pasa a ser el último estado que tenía la (sub) máquina en cuestión
  - **Fork / Join**
    - Misma notación y propósito que las transiciones de las redes de Petri
    - Con un *fork* se "abre" una transición para que pueda terminar en estados de máquinas concurrentes. *Join* hace lo contrario.
  - **Final**
    - Se indica con un círculo que contiene una "T"

Transiciones, con más detalle

- Forma general del rótulo de una transición:

```
nombre_del_evento(parámetros) [condición] / acciones
```

  - Todos los campos son optativos
  - Las acciones pueden ser salidas hacia el exterior, o eventos que otra parte del modelo usa como *trigger*
- Tipos de eventos:
  - **Señal**
    - La recepción de una señal (asincrónica)
    - Es el tipo de evento más común
  - **Tiempo**
    - Se cumplió cierto tiempo desde la llegada al estado
    - Se escribe (ej.) `after 200ms` o también `tm(200ms)`
  - **Llamada**
    - Disparado por otra parte del sistema
  - **Cambio**
    - Cierta expresión condicional pasa a ser cierta
    - Se indica con `when` seguido de la expresión o usando directamente la condición entre corchetes

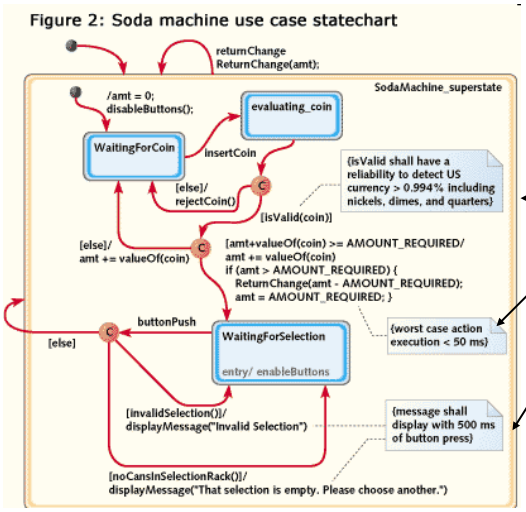
Transiciones, con más detalle

- Forma general del rótulo de una transición:

```
nombre_del_evento(parámetros) [condición] / acciones
```

- La condición
  - La transición se dispara sólo si esa expresión es verdadera
  - Puede incluir (ej.) `in(nombre_de_un_estado)`, para condicionar la transición a que una máquina concurrente con esta esté en cierto estado
    - …como vimos en varios de los ejemplos anteriores
- Ejemplos
  - `evCaenPinos / GenerarRuido()`
  - `evLlegóBola / contBolas++;print("Llegó")`
  - `evCaenPinos(n)[n>=10] / MarcarChuza()`
  - `/ Inicializar()`
  - `[cantidad % 10 == 0]`
  - `after 10ms / InformarTimeout();AbrirPuerta()`
  - `when cantidad % 10 == 0 / foo++;bar++`
  - `Pulsó_Empezar[in(Puerta_Cerrada)] / Arrancar`

Statechart + requisitos de tiempo real



Notas con requermientos de temporización y de calidad de servicio (*quality of service* o QoS)

Source: B.P.Douglass; "Capturing Real-Time Requirements " <http://www.embedded.com/story/OEG20011016S0126>

¿Cómo codificar un Statechart?

- El programa puede ser un ciclo infinito conteniendo ese `switch … case`
  - Para implementar:
    - **Concurrencia**: varios `switch … case` en secuencia
    - **Jerarquías**: `switch … case` anidados
- En lugar de `case` para cada estado se pueden poner los datos del Statechart en un *array* y usar una librería
  - Se pueden guardar punteros a funciones que contengan las condiciones y las acciones
  - Si el `switch … case` quedaba intrincado, esto puede simplificarlo
  - El libro de Samek presenta una librería así
- Programar el Statechart bajo un RTOS
- Usar una herramienta MDD
  - Como IBM Rational Rhapsody o el Real-Time Workshop para Simulink de Mathworks
    - Funcionan con varios RTOS populares

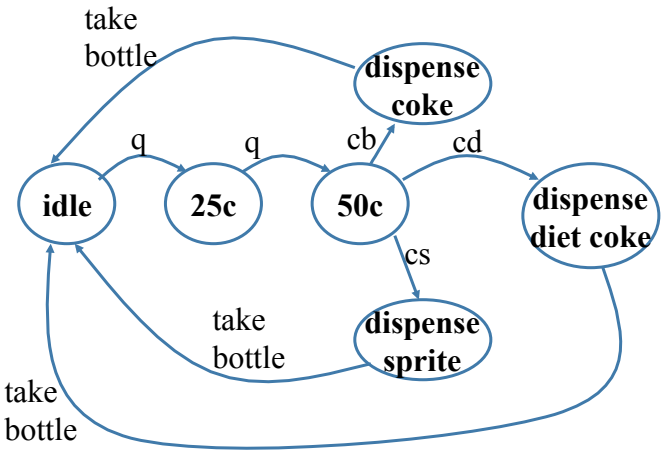
¿Cómo codificar un Statechart?

```
void FSM(int &timer_state, message msg) { // C programs would use int *timer_state
        switch (timer_state) {
                case IDLE_STATE:
                        switch (msg.msg_type) {
                                case START_CMD:
                                        timer.countValue = msg.cmd
                                        timer.start();
                                        timer_state = COUNTING_STATE;
                                        break;
                                default:
                                        // do nothing
                                        break;
                                }; // end switch msg
                        break;
                case COUNTING_STATE:
                        switch (msg.msg_type) {
                                case TIMEOUT:
                                        sw_interrupt(xx);
                                        timer.start();
                                        break;
                                case STOP_CMD:
                                        timer.stop();
                                        timer_state = IDLE_STATE;
                                        break;
                                default:
                                        // do nothing
                                        break;
                                }; // end switch msg
                        break;
                default:
                        cout << "Illegal state value " << endl;
                        break;
        }; // end switch
}; // end FSM function
```

Ejemplo: Expendedor de latas de refresco (1.0)

- Especificaciones de funcionamiento:
  - Cuando se conecte, la máquina espera la introducción de monedas.
  - Cuando se deposite la primera moneda (q), se espera hasta que se deposite la segunda.
  - Una vez depositada, se espera a la selección de refresco.
  - Cuando el usuario pulsa “COKE,” se dispensa una cocacola
  - Cuando el usuario coge la lata, se vuelve al estado de espera
  - Si se elige otro tipo de refresco, se actúa igual.

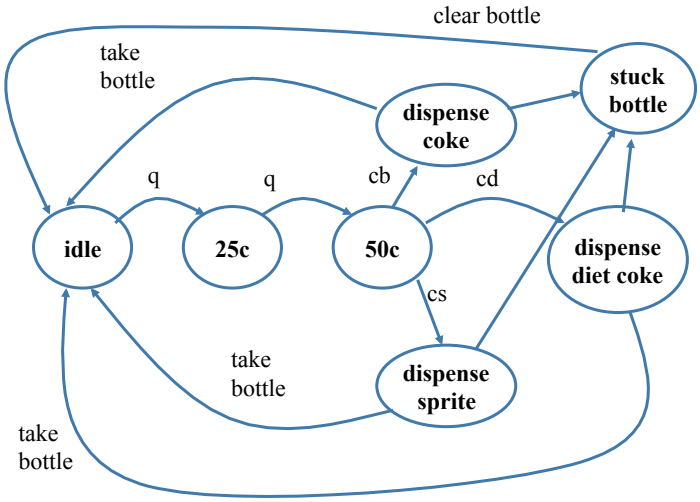
Ejemplo: Expendedor de latas de refresco (1.0)



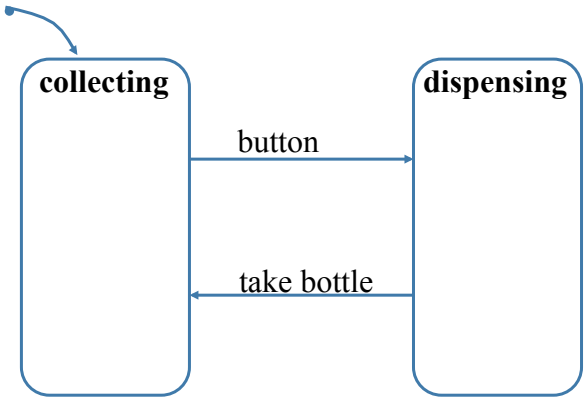
Ejemplo: Expendedor de latas de refresco (2.0)

- Si las latas se atascan:
  - Un indicador se activará notificando esta circunstancia al sistema.
  - En ese caso, no se aceptarán monedas ni se expedirán botes.
  - Cuando se libere el atasco se continuará.

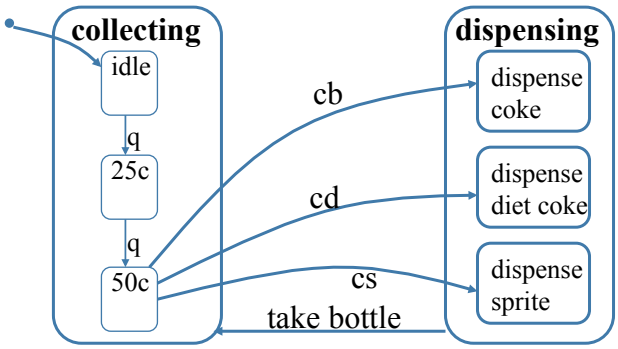
Ejemplo: Expendedor de latas de refresco (2.0)



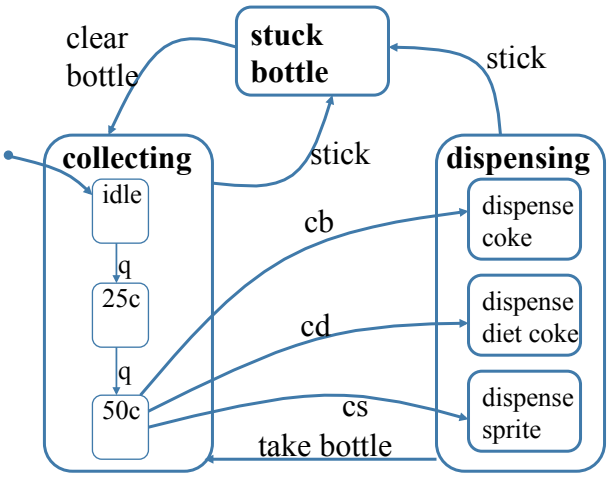
Ejemplo: Expendedor latas refresco, versión Statechart



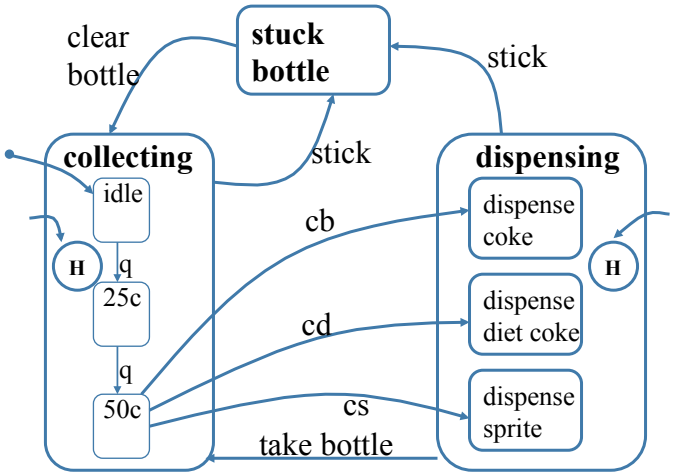
Ejemplo: Expendedor latas refresco, versión Statechart



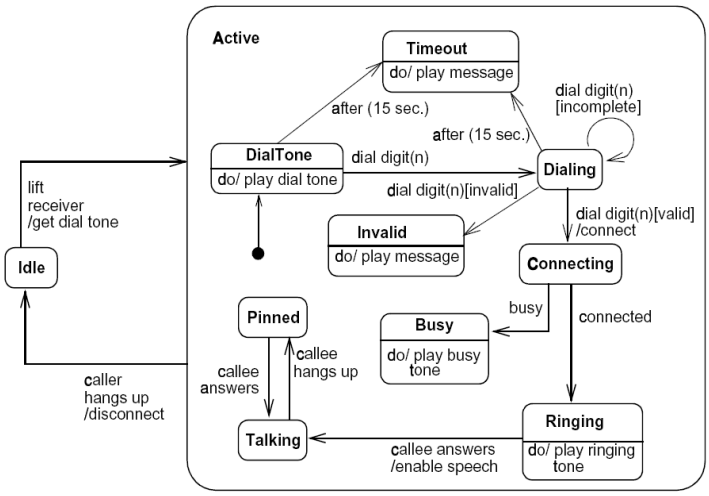
Ejemplo: Expendedor latas refresco, versión Statechart



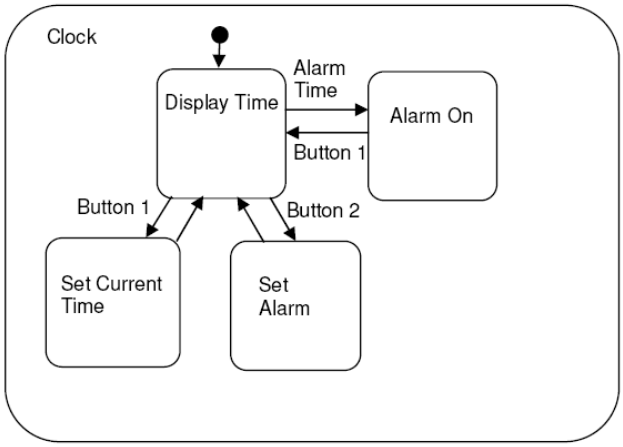
Ejemplo: Expendedor latas refresco, versión Statechart



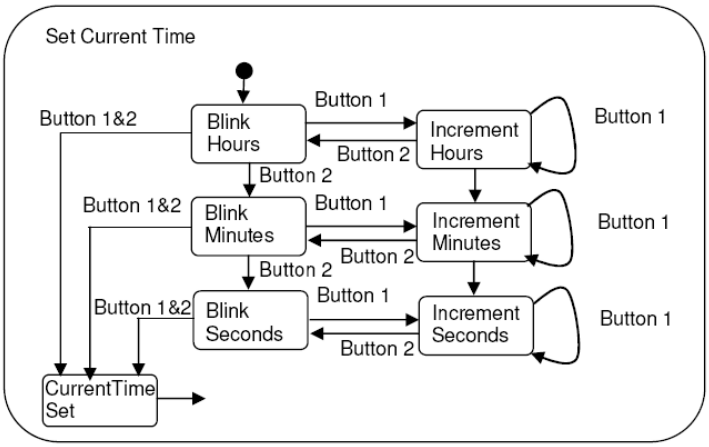
Ejemplo: Teléfono



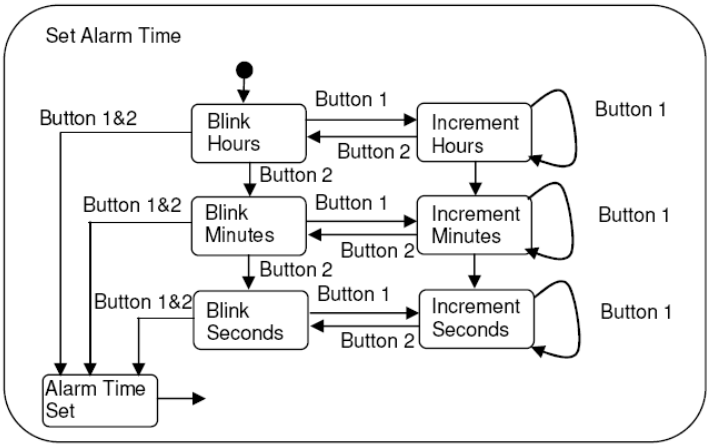
Ejemplo: reloj con alarma



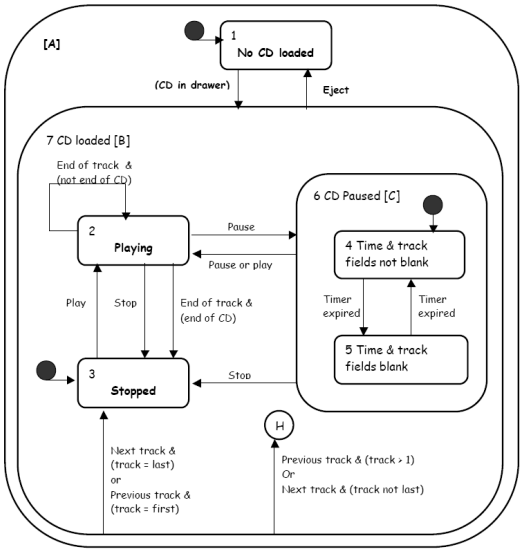
Ejemplo: reloj con alarma



Ejemplo: reloj con alarma



Ejemplo: lector de CD



Ejemplo: lector de CD, modo tabla

Event	Current State			Actions	Next State		
	A	B	C		A	B	C
CD placed in drawer	1	-	-	Close drawer and load CD	7	3	-
Play button pressed	7	3	-	Play CD, show track details on display	7	2	-
Pause button pressed	7	2	-	Pause CD, start pause timer	7	6	4
Pause timer timed out	7	6	4	Blank track and time display	7	6	5
Pause timer timed out	7	6	5	Show track and time display	7	6	4
Pause or play button pressed	7	6	4	Play CD, show track details on display	7	2	-
Pause or play button pressed	7	2	5	Play CD, show track details on display	7	2	-

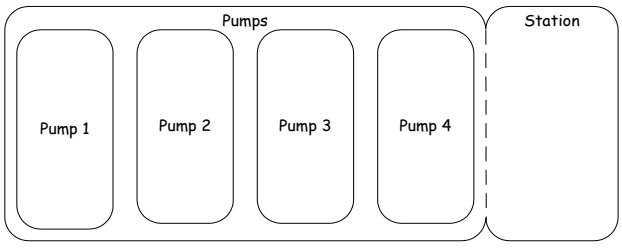
Ejemplo: Gasolinera

1. Cuatro surtidores identificados con un número (pumps).
2. Todos los surtidores son autoservicio.
3. Cada surtidor tiene 3 mangueras: gasolina regular, plus y premium, cada una con un precio propio.
4. Un surtidor solo puede dar servicio a un cliente a la vez.
5. Todos los surtidores están conectados a 3 tanques de almacenamiento, uno por cada tipo de gasolina. Cuando un tanque baja su nivel de almacenamiento de cierto umbral, se activa un aviso para que sea rellenado.
6. Los usuarios pueden repostar mientras que se están rellenando los tanques.
7. Los usuarios deben pagar antes de repostar. Sólo se aceptan pagos en efectivo.
8. Una vez pagado, el empleado activa el surtidor solicitado por el usuario y este puede repostar.
9. El usuario, una vez pagada, puede cambiar el tipo de gasolina que quiere. No podrá hacerlo si ya ha iniciado el repostaje.
10. Si en el coche no cabe toda la gasolina pagada por el usuario, el empleado debe devolverle la diferencia.
11. Cada manguera tiene asociado un sensor para detectar que el tanque del coche está lleno y cortar el suministro.

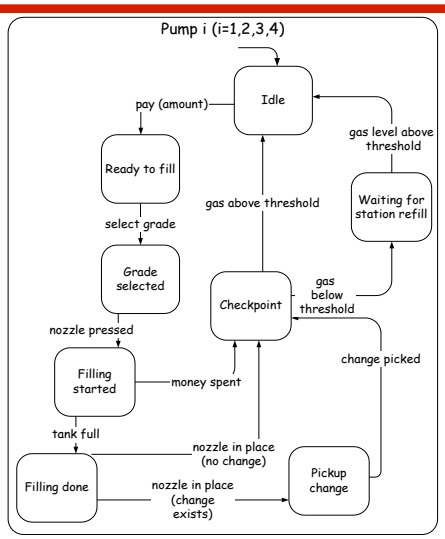
Ejemplo: Gasolinera

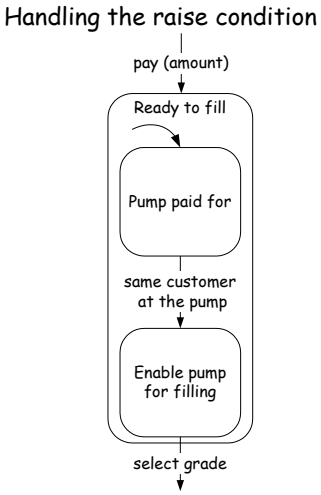
- Eventos
  - "regular", "premium" o "plus"
  - Umbral de gasolina del tanque alcanzado.
  - Pump gas
  - Tanque del coche lleno
  - Pagado
  - Cambio de producto
- Suposiciones
  - Una vez alcanzado el umbral del tanque, queda suficiente gasolina para dar servicio a todos los clientes que están respostando.

Ejemplo: Gasolinera

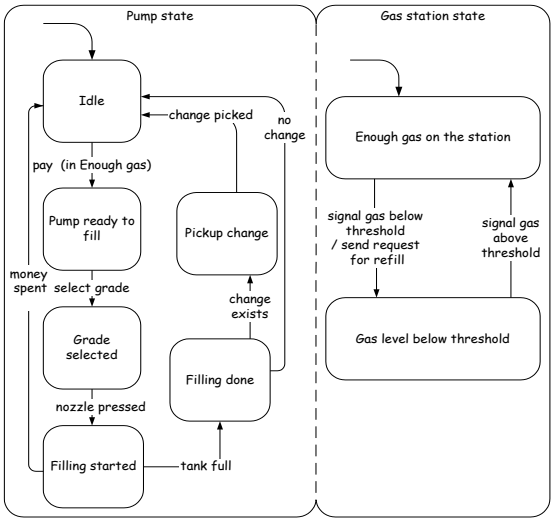


Ejemplo: Gasolinera

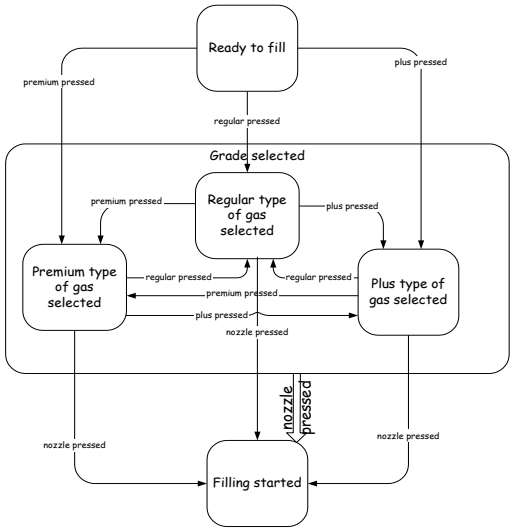




Ejemplo: Gasolinera



Ejemplo: Gasolinera



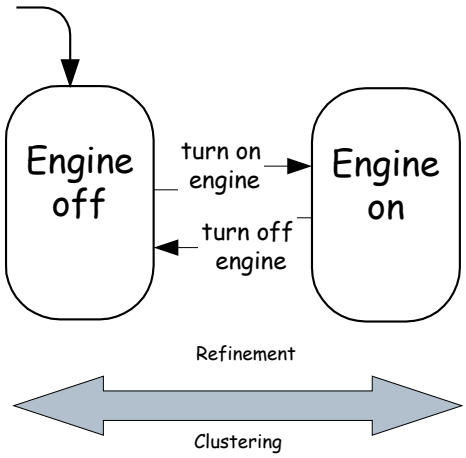
Cruise Control System Specifications

1. You can assume an automatic transmission vehicle.
2. For any of the cruise control (CC) functions to take effect, CC must be turned on first.
3. CC can be in the following states: off, enabled (i.e., on and cruising), and disabled (on, but not cruising).
4. The CC system should be automatically disabled below 30mph and above 90mph.
5. Four actions are permitted during CC: set speed, accelerate, decelerate, and resume speed.
6. When the system is under CC and the brake is pressed, CC is disabled. When the resume button is pressed, the system resumes at the last set CC speed.
7. When the system is under CC and the accelerator pedal is pressed, CC is disabled and the speed increases correspondingly. When the accelerator is released, the CC resumes at its last set CC speed. If at any point of time during acceleration the CC speed is set, CC replaces the old set speed with the new speed.
8. If CC is enabled and the vehicle starts going uphill or downhill, CC should automatically apply the accelerator or brake to maintain the set speed.

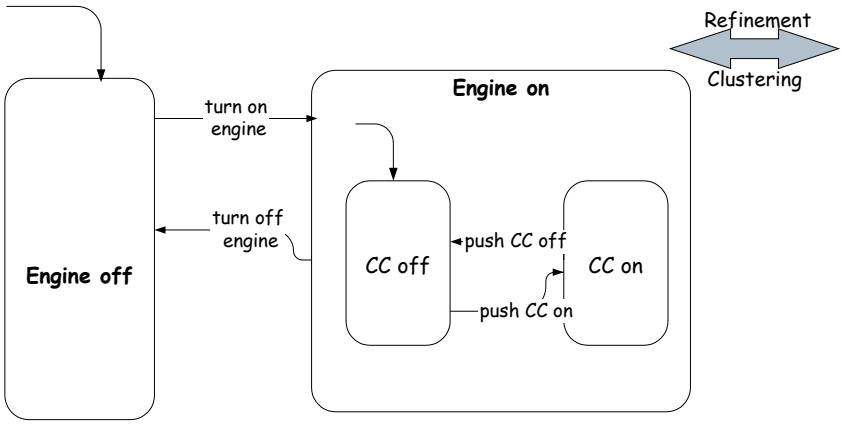
Cruise Control

- Events:
  - Engine on
  - Engine off
  - CC off
  - CC on (+ cruising+ disabled)
  - Set speed (CC is on)
  - Accelerate
  - Decelerate
  - Resume CC

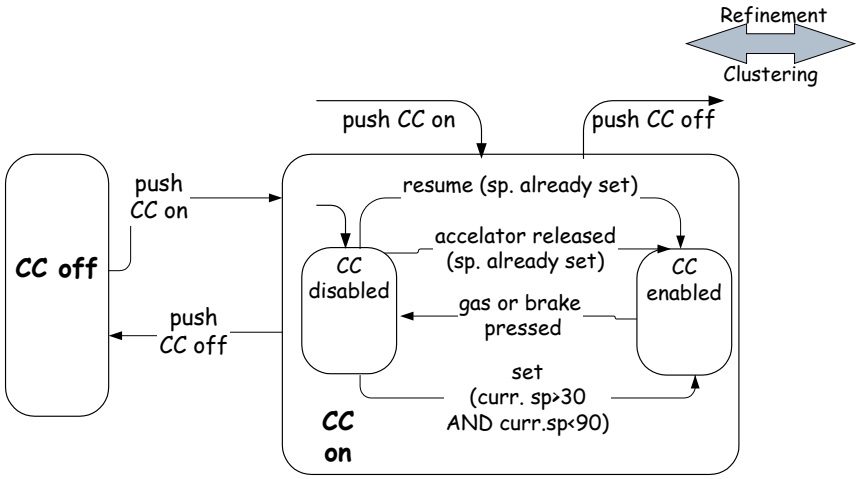
Cruise Control



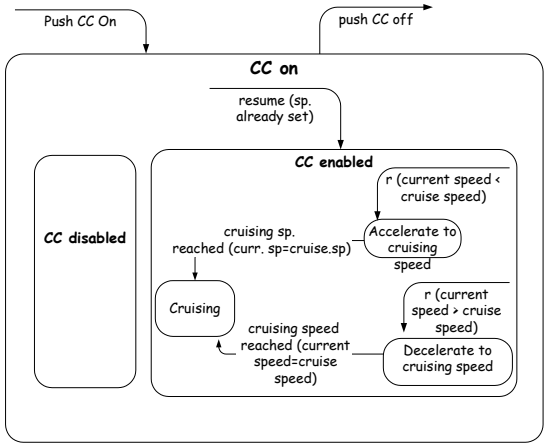
Cruise Control



Cruise Control



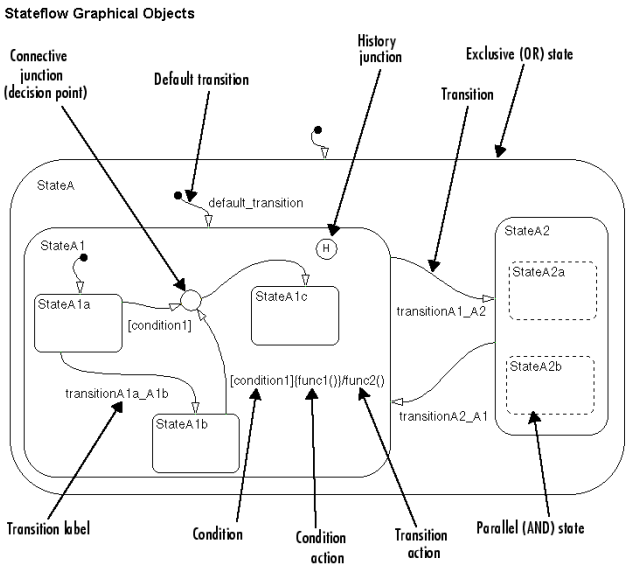
Cruise Control



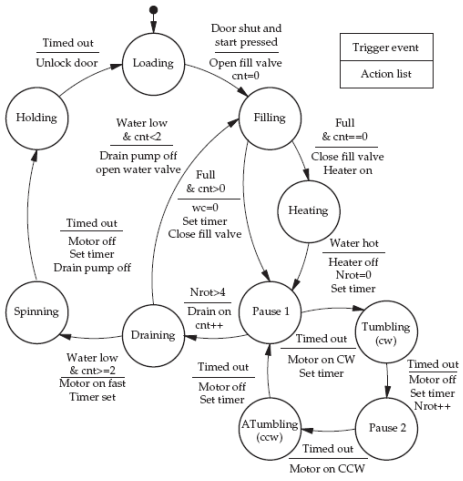
Stateflow

- Se trata de una toolbox de MATLAB para simular FSM y statecharts.
- Integrada con Simulink y el resto de toolboxes de MATLAB.
- Traduce el statechart a código en C.
- Permite combinar elementos gráficos con funciones y otros objetos de MATLAB

Stateflow

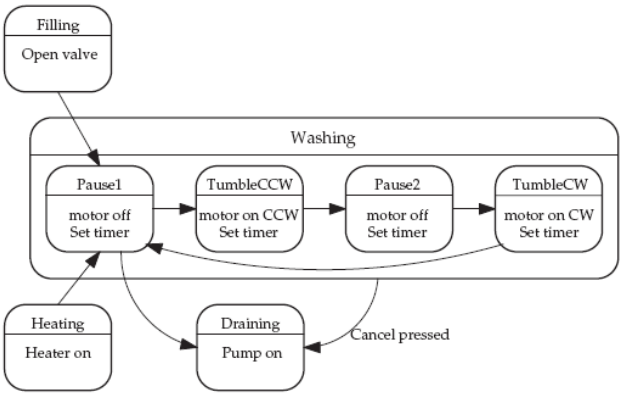


Ejercicio: lavadora con FSM



- Localizar los errores y proponer mejoras.
- Completar el statechart de la siguiente transparencia.

Ejercicio: lavadora con StateChart



Conclusiones

- Los Statecharts son un lenguaje de creciente popularidad en sistemas embebidos
  - Extienden el lenguaje clásico de las máquinas de estados, en una forma que resulta muy práctica para el modelado de software
- Existen diferentes maneras de implementarlos
  - Cada una tiene pros y contras en cuanto a su complejidad, consumo, tiempo de respuesta y necesidad de software de base
- Sirven para detallar las ideas antes de implementarlas, lo que nos sirve para:
  - **Ordenar** las ideas y **documentar**
  - **Ejecutarlos** (o sea, simularlos) para corregir errores y explorar alternativas
  - **Traducirlos automáticamente** a C u otro lenguaje de programación, si contamos con una herramienta MDD apropiada
    - Dentro de estas, normalmente se puede elegir la versión de C, de las librerías y de RTOS (si usamos uno); y se puede trabajar sobre el código resultante