Electrónica Digital

Ingeniería Informática – FICH, UNL
Leonardo Giovanini

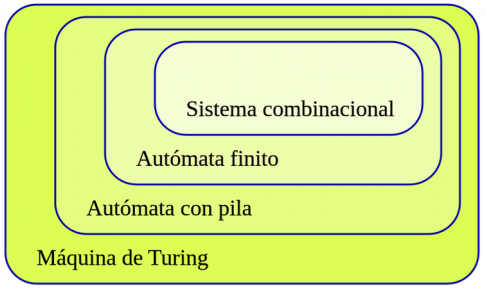
Máquinas de Estados Finitos

Organización de la clase

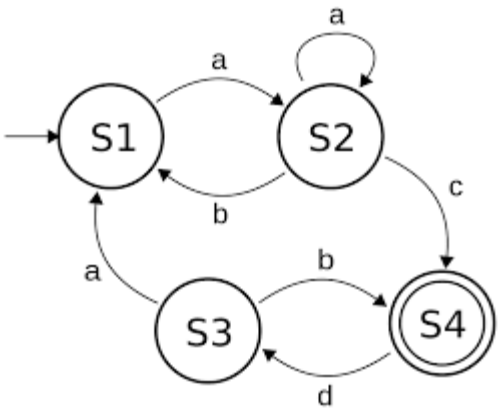
En esta se estudiarán los siguientes temas:

- Conceptos y terminología;
- Definición;
- Representaciones;
- Clasificación;
- Procedimiento de diseño;
- Asignación de estados;
- Descomposición de máquinas;
- Circuitos secuenciales realimentados.

Una ***máquina de estado finito*** (FSM) o simplemente una ***máquina de estado***, es un modelo matemático de computación.
Es una ***máquina abstracta*** que puede estar, en cualquier momento, en ***uno de un número finito de estados***.
El FSM puede cambiar de un estado a otro en respuesta a algunas entradas; ***el cambio de un estado a otro se llama transición***.



Un FSM se define mediante una lista de sus estados, estado inicial y las entradas que desencadenan cada transición. Las máquinas de estados finitos son de ***dos tipos***: ***deterministas*** y ***no deterministas***.



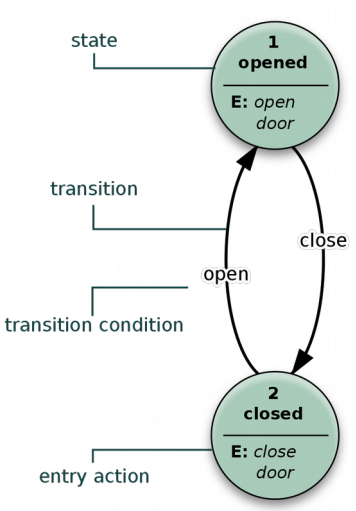
El comportamiento de las FSM se puede observar en muchos dispositivos que realizan una secuencia predeterminada de acciones que dependen de una secuencia de eventos. Ejemplos simples son las ***máquinas expendedoras***, que distribuyen productos; los ***ascendores***, cuya secuencia de paradas está determinada por los pisos solicitados; los ***semáforos***, que cambian la secuencia cuando los automóviles están esperando, y las ***alarmas de seguridad***, que requieren la entrada de una secuencia de números en el orden correcto para desactivarse.

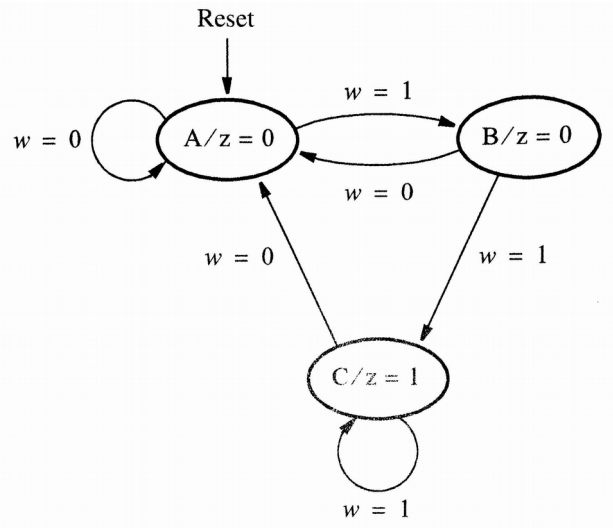
La FSM tiene menos poder computacional los otros modelos de computación. Esto significa que ***hay tareas computacionales que una máquina de Turing puede hacer pero una FSM no***. Esto se debe a que la memoria de un FSM está limitada por el número de estados que tiene.
Los FSM se estudian en el campo más general de la teoría de autómatas.

Un ***estado*** es una descripción de ***las variables internas*** de un sistema, situacion, que está esperando ejecutar una transición.
Depende del ***estado inicial*** y las ***entradas anteriores***.

Dentro de los estados puden haber ***estados finales***, que se utilizan cuando finaliza el procesamiento de modo que puede o no generar una salida y ***detener la FSM***.

Por ejemplo, las puertas automáticas en los supermercados tienen dos estados: cerrado o abierto; o los semáforos contienen tres estados: rojo, amarillo, verde.
Los ejemplos más sofisticados pueden contener muchos estados, pero nunca un número infinito. No tener estados tampoco tiene sentido.





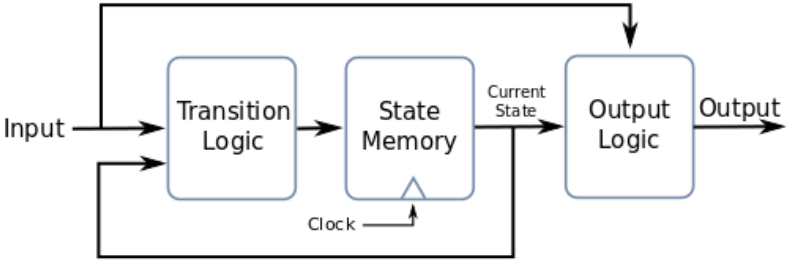
Una ***transición*** es un ***conjunto de acciones*** que se ejecután cuando se ***cumple una condición*** o cuando se ***recibe un evento***.

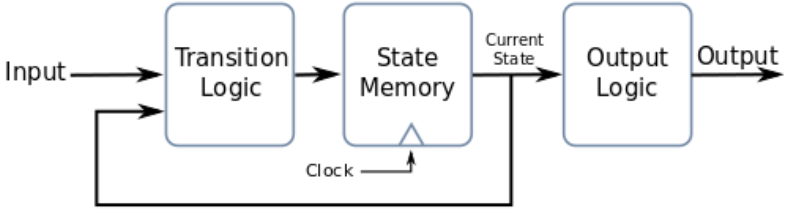
Cada transición contiene un conjunto de símbolos.
El conjunto de todos los símbolos presentados en el FSM se llama alfabeto. Cuando el estado de origen está activo y se ejecuta la transición, cambia el estado activo del estado de origen al estado de destino solo si la entrada actual coincide con el símbolo de la transición.

Una máquina de estados finito esta definido por un sextuple $(\Sigma, \Gamma, \mathcal{S}, s_0, T, G)$ donde:

- Σ es el alfabeto de entrada, un conjunto finito y no vacío de símbolos;
- Γ es el alfabeto de salida, conjunto finito y no vacío de símbolos;
- $\mathcal{S}$ es un conjunto finito y no vacío de estados, que incluye el estado inicial $s_0 \in \mathcal{S}$ y al conjunto de estados finales $\mathcal{F}$, el cual puede vacío;
- $T: \mathcal{S} \times \Sigma \to \mathcal{S}$ es la función de transición de estado. En una FSM no determinista estaria definida por $T: \mathcal{S} \times \Sigma \to \mathbb{P}(\mathcal{S})$, es decir devolverá un conjunto de estados;
- G es la función de salida.

Si la función de salida ***depende del estado y la entrada*** $G: \mathcal{S} \times \Sigma \to \Gamma$, esa definición corresponde al modelo Mealy.





Si la función de salida ***solo depende del estado*** $G: \mathcal{S} \to \Gamma$, corresponde al ***modelo de Moore***.

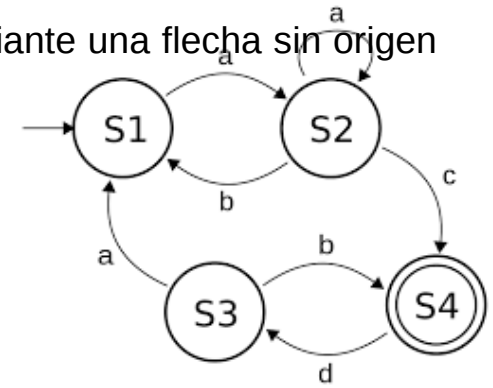
Una ***FSM sin función de salida*** se conoce como ***semiautomata*** o sistema de transición.

Un ***diagrama de estado*** describe el ***comportamiento*** de los sistemas por una serie de eventos que pueden ocurrir en uno o más estados posibles.

Una forma clásical de diagrama de estado es un ***grafo orientado*** con elementos que representan los componentes de las FSM:

Vértices Q – representan los estados $s_i \in \mathcal{S}$ a traves de círculos y etiquetados con símbolos o palabras únicos dentro de ellos. Estas etiquetas representan el estado representado. Los estados finales se dibujan con un doble circulo.

El estado inicial s_0 no suele representarse con un vertice, se lo representa mediante una flecha sin origen que apunta al siguiente estado.



Arcos $t(q_i, \sigma_j) \in T$ – representan las transiciones entre estados causadas por la entrada σ_j, identificada en los bordes. Un arco se ***dibuja como una flecha dirigida*** desde el estado actual al siguiente, describiendo que la transicion que ocurrira si el símbolo de entrada aparece. Este símbolo y el valor correspondiente aparecen junto al arco que representa la transición.

En una ***máquina Mealy*** la entrada y la salida están representadas en cada borde separadas con una barra "/": 1/0 denota el cambio al encontrar el símbolo 1, lo que hace que se emita el símbolo 0.

Para una ***máquina Moore***, la salida del estado generalmente se escribe dentro del círculo del estado, también separada con una barra /. También hay variantes que combinan estas dos notaciones.

Una ***tabla de transición de estados*** es una tabla que muestra a qué estado (o estados en el caso de un autómata finito no determinista) se moverá una FSM, en función del estado actual y las entradas.
Es esencialmente una tabla de verdad en la que las entradas incluyen el estado actual junto con otras entradas, y las salidas incluyen el siguiente estado junto con otras salidas.

Input	Current state	Next state	Output
I_1	S_1	S_i	O_x
I_2	S_1	S_j	O_y
…	…	…	…
I_n	S_1	S_k	O_z
I_1	S_2	$S_{i'}$	$O_{x'}$
I_2	S_2	$S_{j'}$	$O_{y'}$
…	…	…	…
I_n	S_2	$S_{k'}$	$O_{z'}$
…	…	…	…
I_1	S_m	$S_{i''}$	$O_{x''}$
I_2	S_m	$S_{j''}$	$O_{y''}$
…	…	…	…
I_n	S_m	$S_{k''}$	$O_{z''}$

Las tablas de transición son a veces tablas unidimensionales parecidas a las tablas de verdad. La única dimensión indica entradas, estados actuales, estados siguientes y salidas asociadas con las transiciones de estado.

Next state / Current state	S_1	S_2	…	S_m
S_1	I_i/O_x	—	…	—
S_2	—	—	…	I_j/O_y
…	…	…	…	…
S_m	—	I_k/O_z	…	—

Input / Current state	I_1	I_2	…	I_n
S_1	S_i/O_x	S_j/O_y	…	S_k/O_z
S_2	$S_{i'}/O_{x'}$	$S_{j'}/O_{y'}$	…	$S_{k'}/O_{z'}$
…	…	…	…	…
S_m	$S_{i''}/O_{x''}$	$S_{j''}/O_{z''}$	…	$S_{k''}/O_{z''}$

Las tablas de transición de estado son típicamente tablas bidimensionales. Hay dos formas comunes de organizarlos.

En la primera forma una de las dimensiones indica estados actuales, mientras que la otra indica entradas. Las intersecciones de fila/columna indican los siguientes estados y salidas asociadas con las transiciones.

En la segunda forma, una de las dimensiones indica los estados actuales, mientras que la otra indica los siguientes. Las intersecciones de fila/columna indican entradas y salidas asociadas con las transiciones.

El UML es un ***lenguaje gráfico*** para visualizar, especificar, construir y documentar un sistema.
Se trata de organizar la forma en que funciona un sistema, o cada subsistema, de manera que esté siempre en uno de los estados posibles y haya transiciones condicionales bien definidas entre los estados.

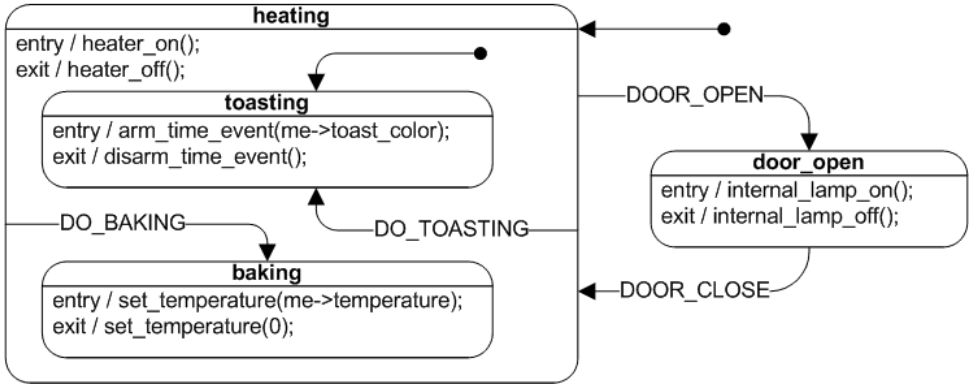
Las FSM basadas en UML ***superan las limitaciones*** de las FSM y ***conservan sus principales beneficios***.

Conservan la forma general de los diagramas de estado tradicionales e introducen los nuevos conceptos:

Estados jerarquicos anidados − los estados pueden contener otros estados de modo que ***tengan una estructura interna*** (estados compuestos). Esta estructura puede ser arbitrariamente compleja de modo que se genera una estructura jerarquica. La descomposición jerárquica está diseñada para facilitar la reutilización del comportamiento, heredando el comportamiento de un estado superior a partir de ignorar los eventos manejados de manera común. El anidamiento de estado jerárquico permite la programación por diferencia; y

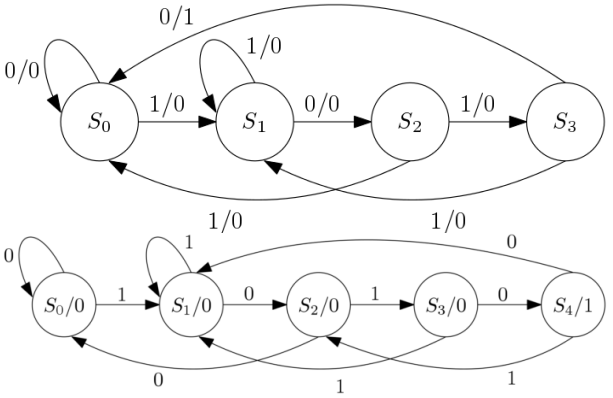
Descomposición AND-complementaria − significa que un estado compuesto puede contener dos o más subsistemas compatibles e independientes (regiones ortogonales) que permiten fragmentar el sistema en partes independientes y activas simultáneamente. Esta descomposición permite la mezcla de comportamientos independientes como producto cartesiano para que permanezcan separados.

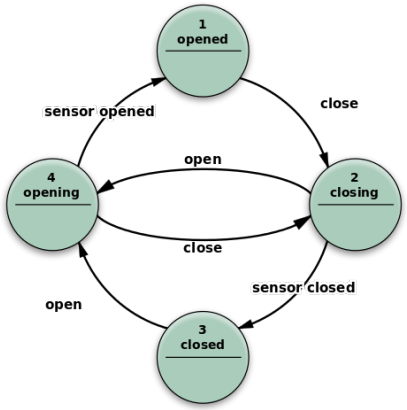
Las FSM-UML tienen las características tanto de las máquinas Mealy como las de Moore. Admiten acciones que dependen tanto del estado del sistema como del evento desencadenante (máquinas Mealy), así como acciones de entrada y salida que están asociadas con estados en lugar de transiciones (máquinas Moore).



Las máquinas de estado finito se pueden clasificar en:

Receptores − también llamados detectores, producen una salida binaria que indica si se acepta o no la entrada recibida. Cada estado de un detector es aceptar o no aceptar. Una vez que se ha recibido toda la secuencia, si el estado actual es un estado de aceptación, se acepta la entrada; de lo contrario es rechazado. No se utilizan acciones.



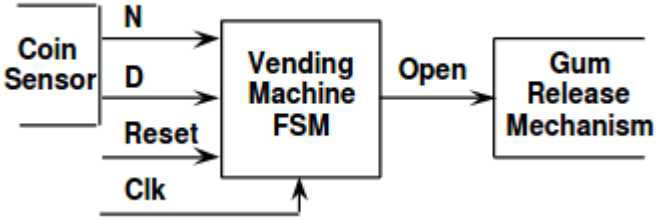


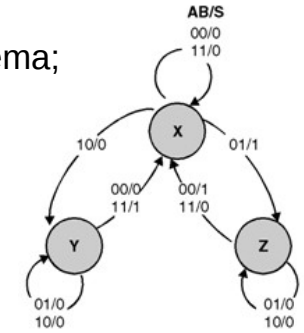
Transductores − es un FSM que produce salida y entrada de lectura. Consiste en un número finito de estados que están unidos por transiciones etiquetadas con un par de entrada / salida. El FSM comienza en un estado de inicio designado y salta a diferentes estados dependiendo de la entrada, mientras produce salida de acuerdo con su tabla de transición. Tienen buenas propiedades algebraicas, se pueden combinar libremente (formar un álgebra) bajo composición, lo que implementa la composición relacional en las relaciones regulares (piense en esto como composición de funciones) mientras se mantiene muy compacto.

Secuenciadores − también llamados generadores, son una subclase de receptores y transductores que tienen un alfabeto de entrada de una sola letra. Producen solo una secuencia que se puede ver como una secuencia de salida de salidas de un transductor

Modelado del problema

1. Dibuje un diagrama de bloques del sistema o pseudocodigo;
2. Identifique las entradas y salidas;
3. Dibuje un diagrama de estado que represente al sistema;
4. Minimice el numero de estados;
5. Obtenga la tabla de transicion de estados.





PS	NS			
	AB			
	00	01	11	10
x	x/0	z/0	x/0	y/0
y	x/0	y/0	X/1	y/0
z	x/0	z/0	x/0	z/0

Diseño de la FSM

1. Codificar los estados
2. Para una maquina Moore — Reescribir la tabla de transicion con la codificacion seleccionada y escribir la tabla de salida;
3. Para una maquina Mealy — Reescribir la tablas de transicion y salida con la codificacion seleccionada;
4. Elija el tipo de flip-flop a utilizar y obtenga las tablas de excitacion;
5. Obtenga las tablas de verdad de las funciones logica de transicion de estados y salida;
6. Diseñe el la funcion logica de transicion de estados y salida;

Implementacion de la FSM

1. Dibuje el esquematico;
2. Implemente el sistema utilizando un lenguaje de descripcion de h