Introducción a UML y Statecharts.

Juan Alarcón.
jalarcon@frba.utn.edu.ar

Agenda.

- ¿Qué es UML?¿Para qué sirve?
- Áreas de UML.
- Máquinas de estados finitos (FSM)
  - Definiciones
  - Ejemplo.
  - Diseño con FSM
- Statecharts
- Herramientas.
- Ejemplo.
- Bibliografía.

¿Qué es UML?

- El Lenguaje unificado de modelado (UML) es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de software.
- Es un lenguaje para modelado discreto de propósito general. Se utiliza en software, firmware y lógica digital.
- Es un lenguaje estandarizado (ISO/IEC 19501:2005 Information technology — Open Distributed Processing)
- Está fuertemente relacionado con la programación orientada a objetos

Objetivos de UML.

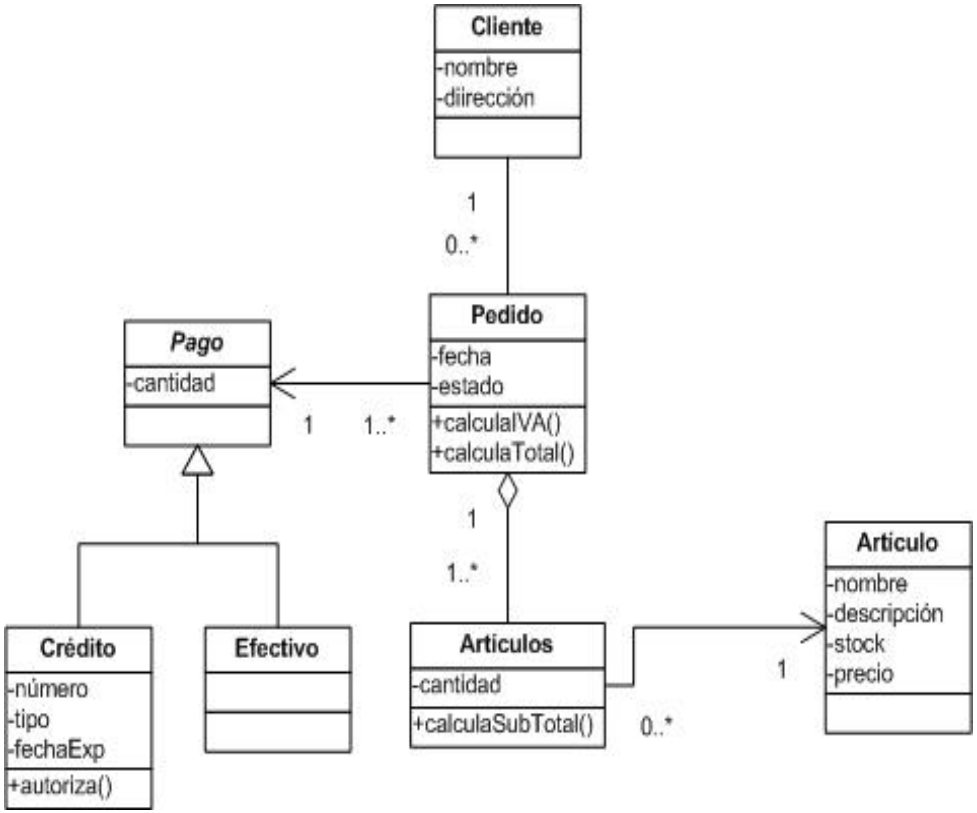
- Lenguaje de modelado de sistemas discretos de propósito general.
- Pretende abordar problemas de desarrollo de software tales como: complejidad, patrones, concurrencia y desarrollo en equipo.
- UML es una herramienta, no un método de desarrollo.

Áreas de UML

- ***Estructura estática***. Describe un conjunto de objetos discretos que almacenan información y se comunican para implementar un comportamiento que modelan al sistema. También describe las relaciones entre objetos.
- ***Comportamiento dinámico***. La historia de la vida del objeto (máquina de estados) y la forma que interactúa con otros objetos (colaboración).
- ***Construcciones de implementación***.
- ***Organización del modelo***. Estructura jerárquica por medio del uso de paquetes.
- ***Mecanismos de extensión.***

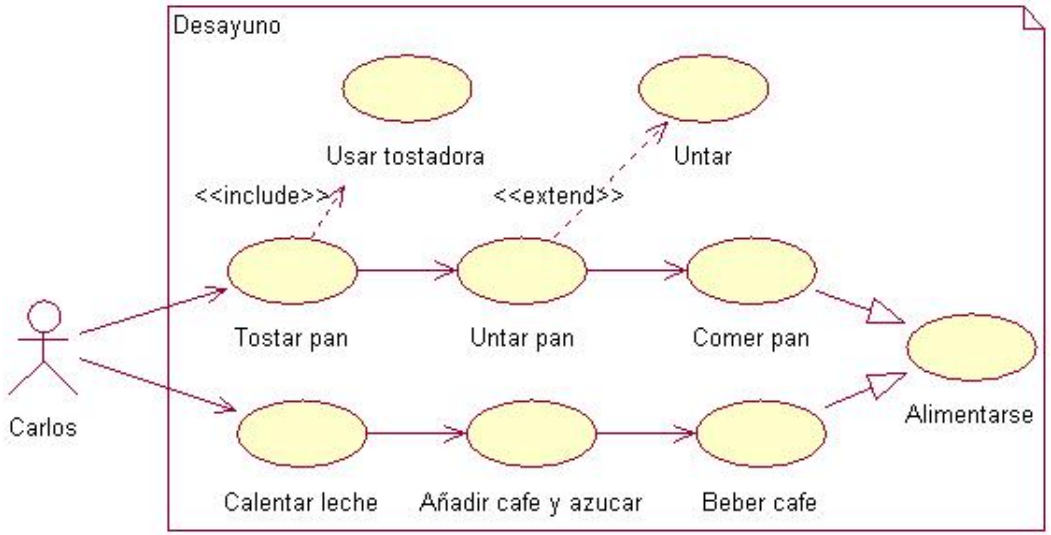
Área estructural. Vista estática.

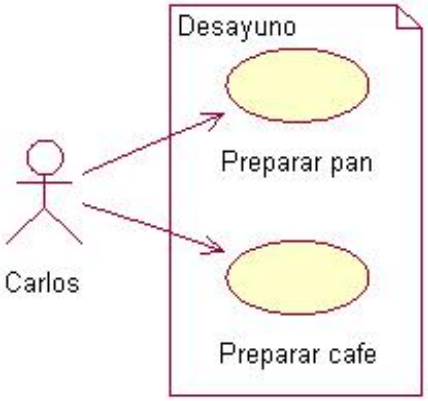
- ***Vista estática.*** Sus componentes principales van a ser las clases y sus relaciones.



Área estructural. Casos de uso.

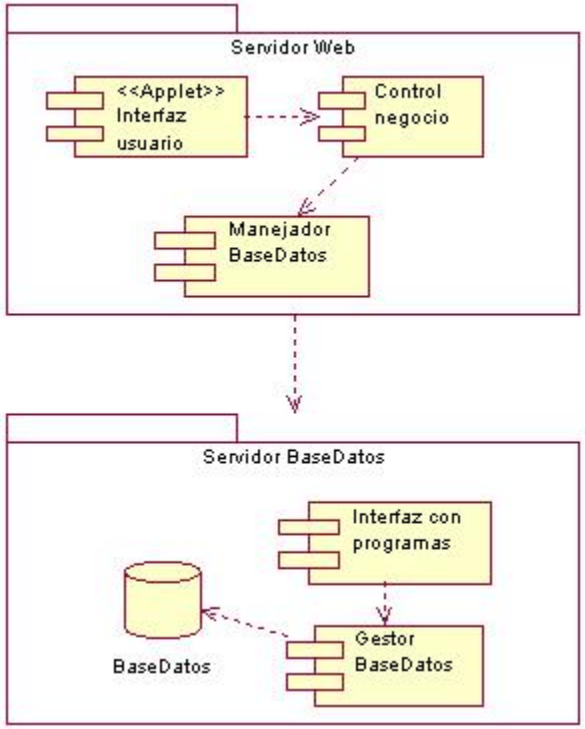
- ***Vista de los casos de uso.*** Modela la funcionalidad del sistema según lo perciben los usuarios externos





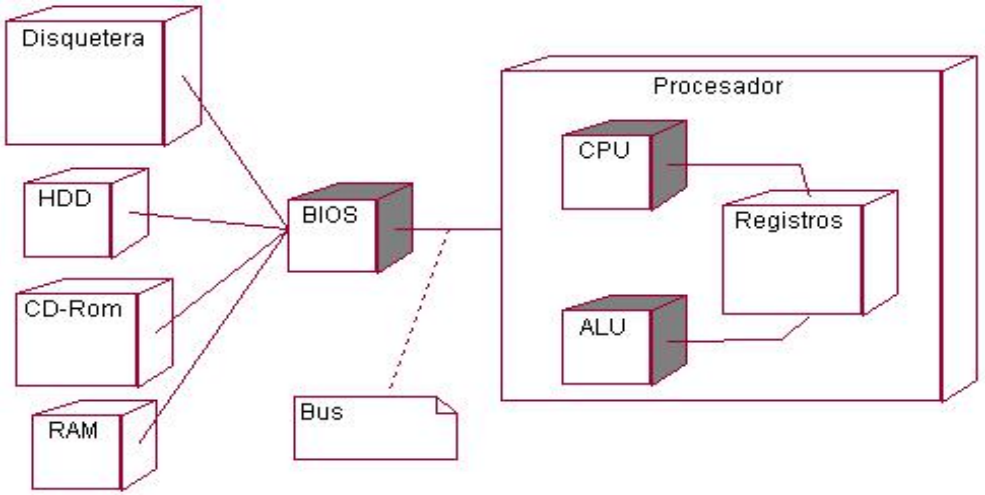
Área estructural. Diagramas de componentes.

- Los componentes son módulos de código. Los diagramas de componentes muestran como está organizado un conjunto de componentes y las dependencias que existen entre ellos.



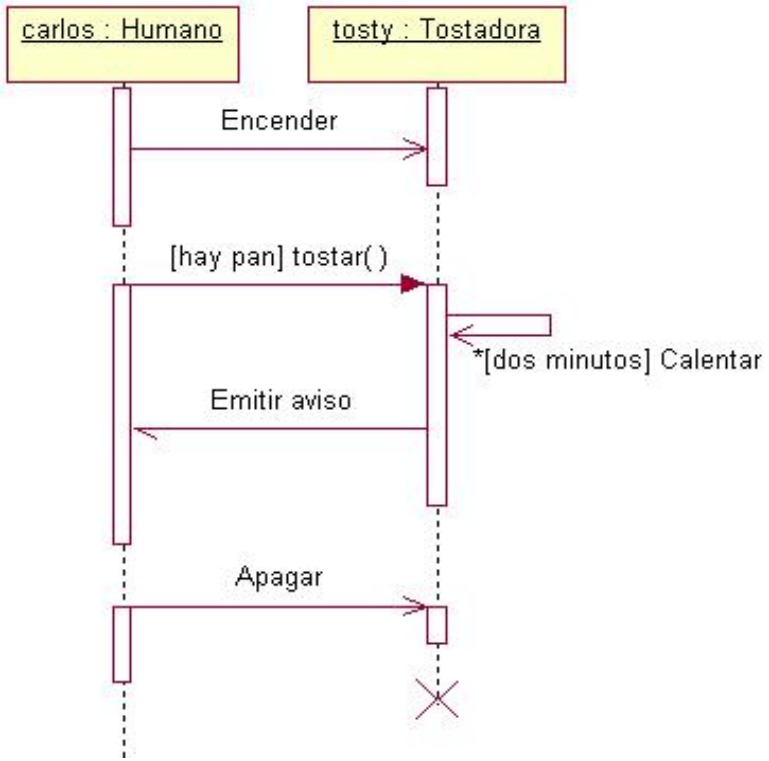
Área estructural. Diagramas de despliegue.

- Representa la disposición de las instancias de componentes de ejecución en instancias de nodos. Un nodo es un recurso en ejecución.



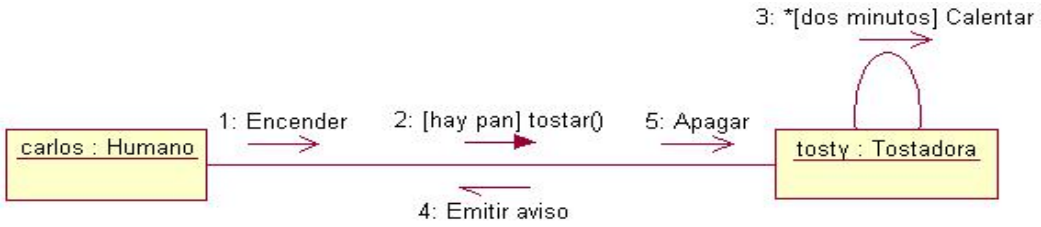
Comportamiento dinámico. Diagramas de secuencia.

- Los diagramas de secuencia describen como los objetos del sistema colaboran. Se trata de un diagrama de interacción que detalla como las operaciones se llevan a cabo, qué mensajes son enviados y cuando, organizado ***todo en torno al tiempo.***



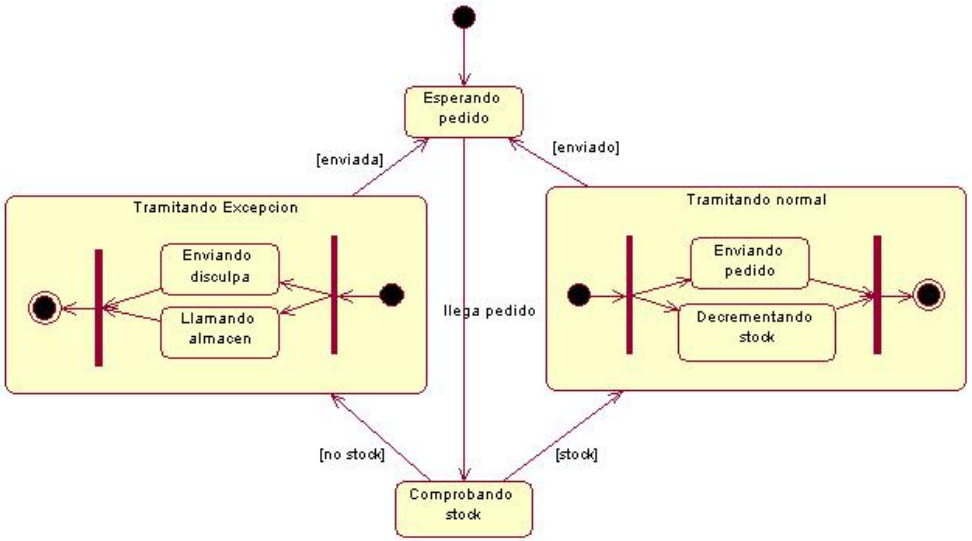
Comportamiento dinámico. Diagramas de colaboración.

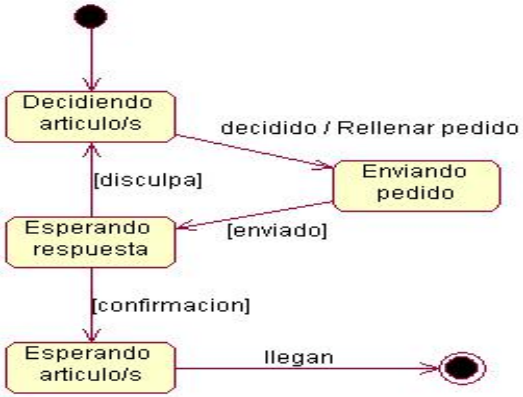
- Los diagramas de colaboración son otro tipo de diagramas de interacción, que contiene la misma información que los de secuencia, sólo que se centran en las ***responsabilidades de cada objeto***, en lugar de en el tiempo en que los mensajes son enviados



Comportamiento dinámico. Diagramas de estados.

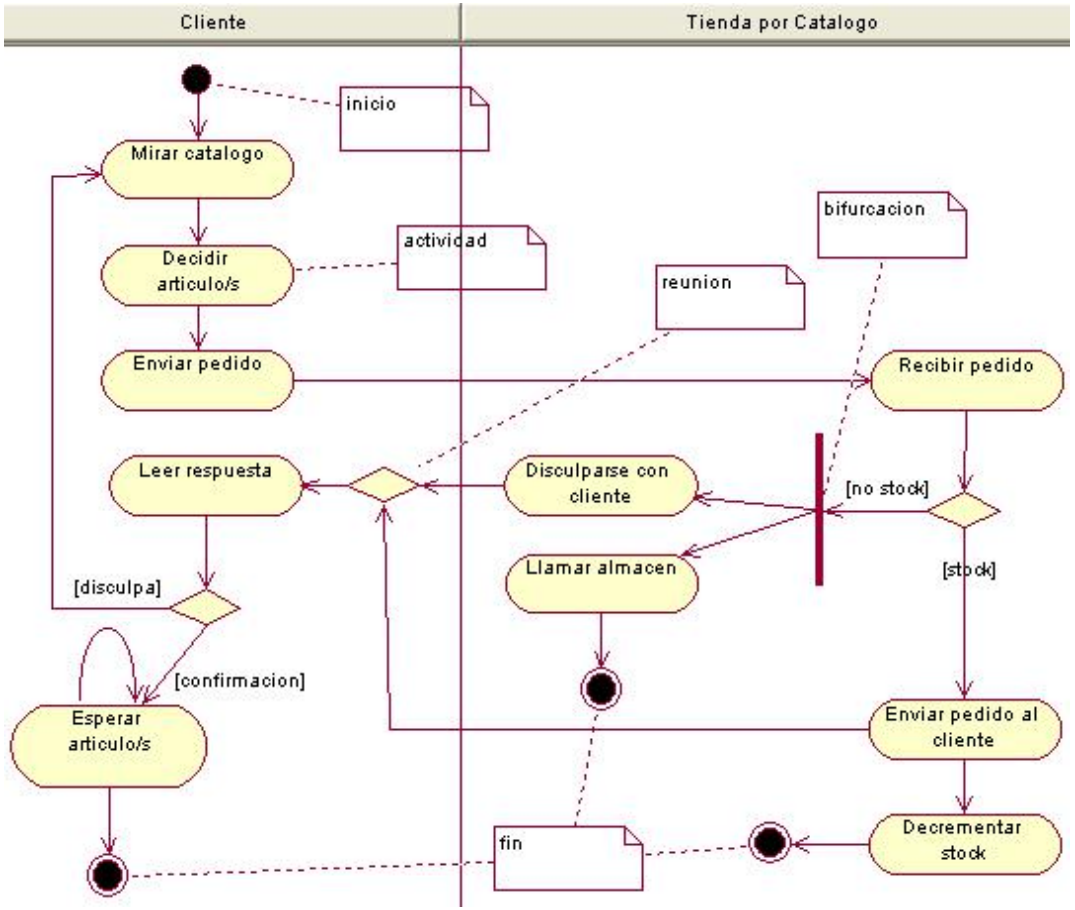
- Los diagramas de estados muestran los posibles estados en que puede encontrarse un objeto y las transiciones que pueden causar un cambio de estado.





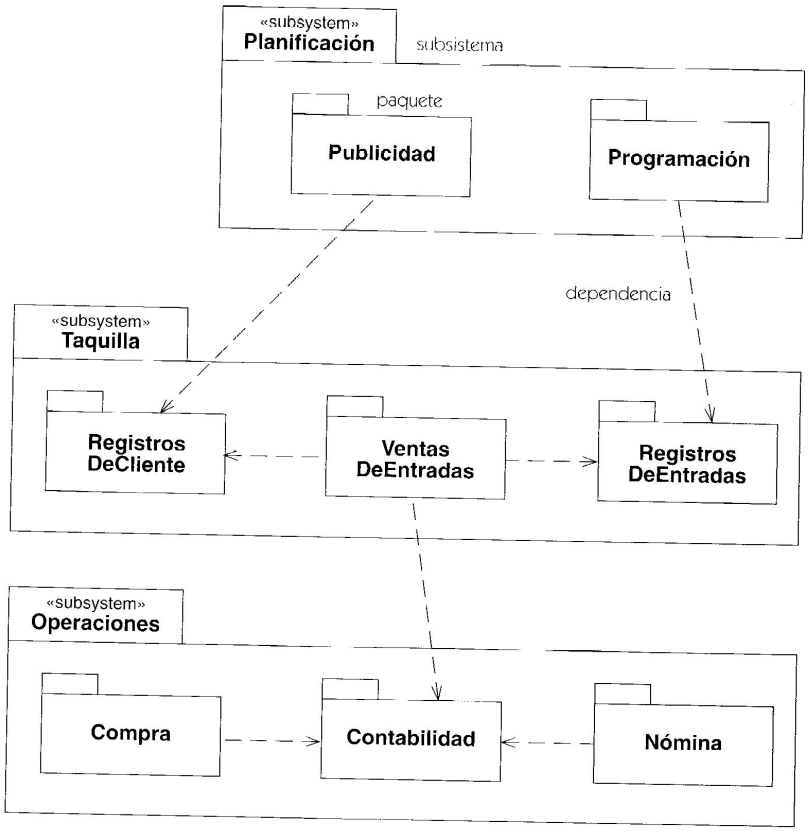
Comportamiento dinámico. Diagramas de actividades.

- Los diagramas de actividades son básicamente ***diagramas de flujo adornados***, que guardan mucha similitud con los diagramas de estados. Mientras que los diagramas de estados centran su atención en el proceso que está llevando a cabo un objeto, los ***diagramas de actividades muestran como las actividades fluyen y las dependencias entre ellas***



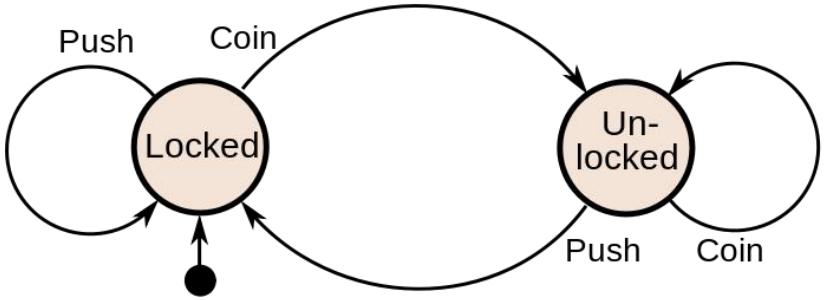
Organización del modelo. Paquetes y Subsistemas.

- Se puede considerar a los paquetes como “contenedores” de elementos de UML (máquinas de estados, diagramas de clase, otros paquetes).
- Un subsistema es un paquete especial, con una interfaz totalmente definida.



Máquina de Estados Finitos

- Una máquina de estados es un ***modelo de comportamiento*** de un sistema, donde la(s) salida(s) dependen de las entradas anteriores y actuales.
- Ejemplo: Un molinete.



Máquinas de estados finitos (FSM). Definiciones.

- ***Estado:*** Situación o condición de un sistema, donde éste realiza alguna actividad o espera algún evento externo.

- ***Máquina de estados finitos (FSM):*** Sistema que puede ser particionado en un conjunto acotado de estados que no se solapan.

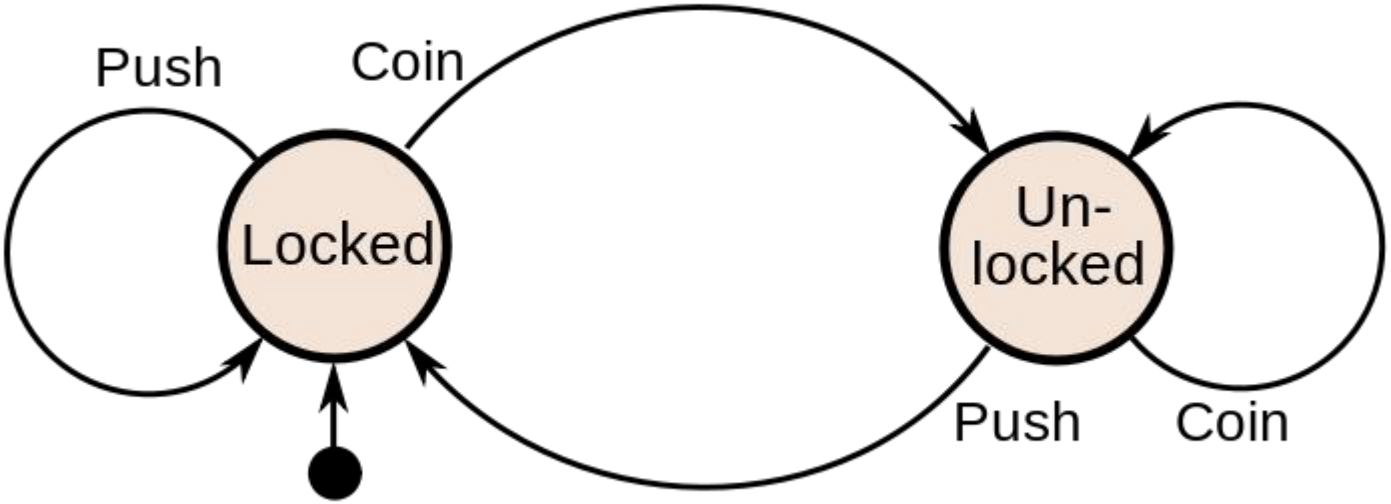
Máquinas de estados finitos (FSM). Definiciones.

- ***Evento:*** Es la ocurrencia en tiempo y espacio de algo significativo para el sistema.

- ***Acción:*** Es la respuesta del sistema a un cambio de estado, evento o cualquier cambio de interés para el sistema.

- ***Transición:*** Es el cambio entre un estado y otro

Reconociendo componentes



Clasificación de máquinas de estados finitos.

- ***Reconocedoras:*** Son aquellas que tienen una salida binaria que depende únicamente del estado y que genera una salida positiva luego de una secuencia de entrada determinada (TD1).

- ***Transductoras:*** Convierten una secuencia de señales de entrada en una secuencia de salida, pudiendo ser una salida binaria o más compleja.

Clasificación de máquinas de estados finitos (2).

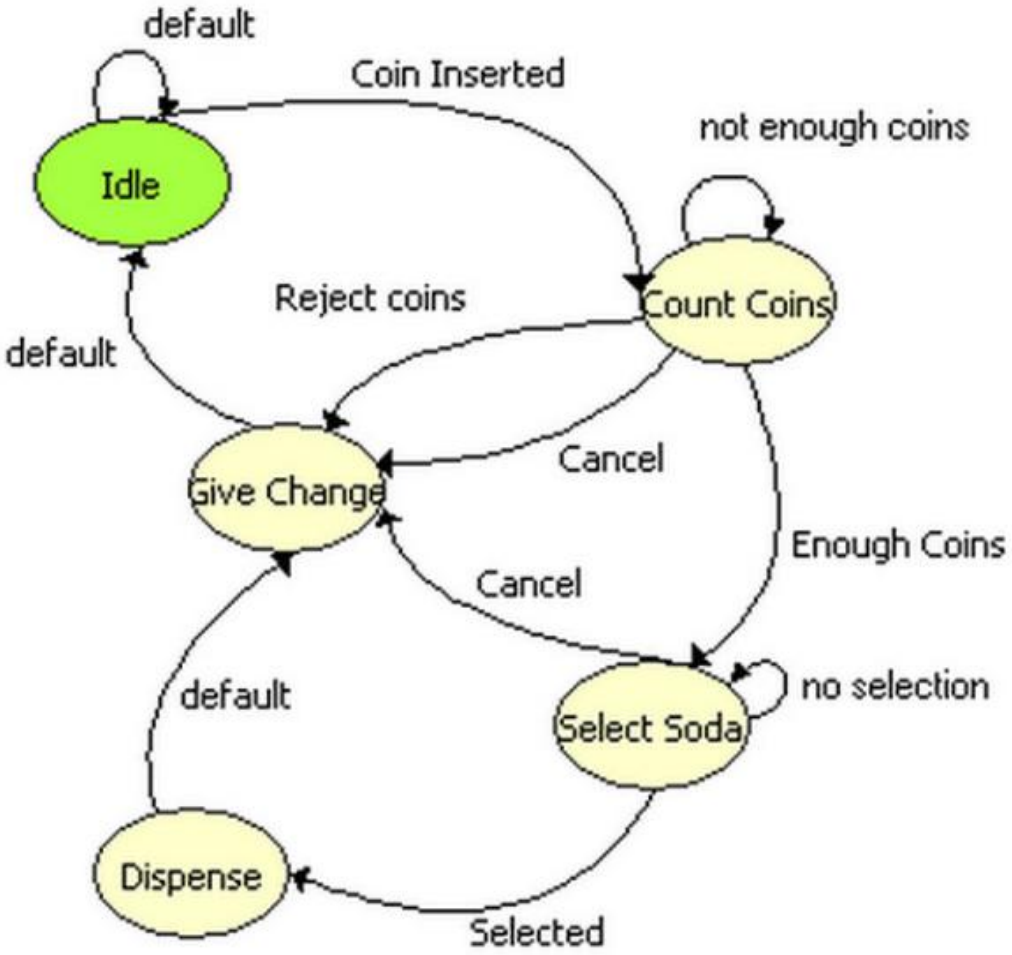
- ***Máquina de Moore:*** Las salidas actuales de la máquina de estados dependen exclusivamente del estado actual del sistema.

- ***Máquina de Mealy:*** Las salidas actuales de la máquina de estados dependen de las entradas actuales y del estado actual.

Diseñando con FSM.

- Se desea construir una máquina de vender bebidas que:
  - Espera una cantidad determinada de monedas para entregar la bebida y el vuelto.
  - Permita elegir el tipo de bebida (todas con el mismo costo)
  - Tenga un botón para cancelar la compra y entregar el cambio.

Diseñando con FSM.



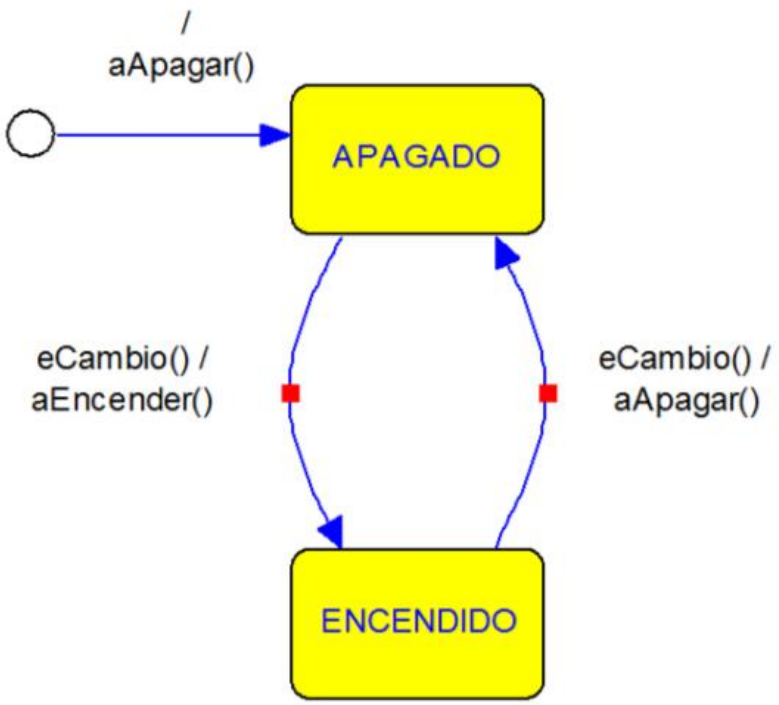
Resultados del diseño con FSM

- Complejidad y máquinas de estados. Con un bajo aumento de la complejidad, la cantidad de estados y transiciones aumenta casi exponencialmente “explosión de estados”
- ¿Cómo atacar a la complejidad?
  - Jerarquía
  - Modelos más completos.
- Uso de statecharts.

Statecharts.

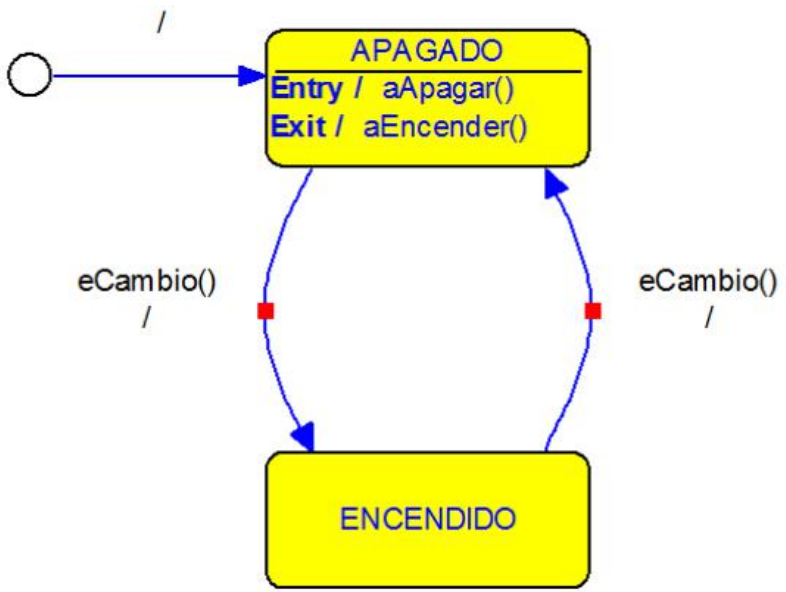
- Los statecharts son una extensión de las máquinas de estados. Estas definen:
  - Acciones de Entrada y Salida
  - Guardas.
  - Estados jerárquicos.
  - Concurrencia
  - Transiciones internas.
  - Pseudoestados

Estados en Statecharts



- Eventos
  - ***eCambio()***
- Acciones
  - ***aApagar()***
  - ***aEncender()***
- Estados
  - ***Apagado***
  - ***Encendido***

Estados. Acciones de entrada y salida.

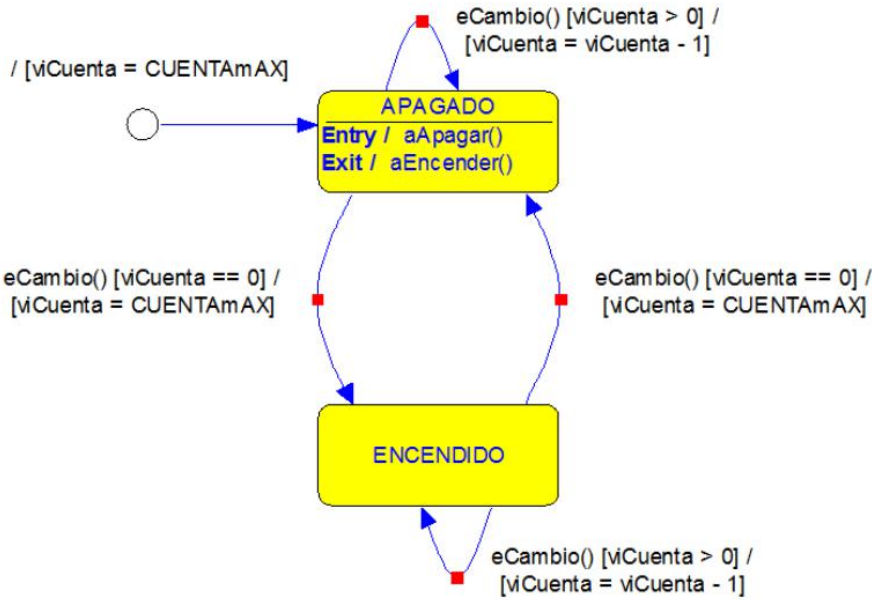


- Las acciones Entry y Exit se ejecutan cuando se entra a un estado y cuando se sale del mismo.
- Son acciones opcionales.
- ***Se ejecutan siempre que se entra o se sale del estado, no importa por que transición se realice.***

Estados. Acciones.

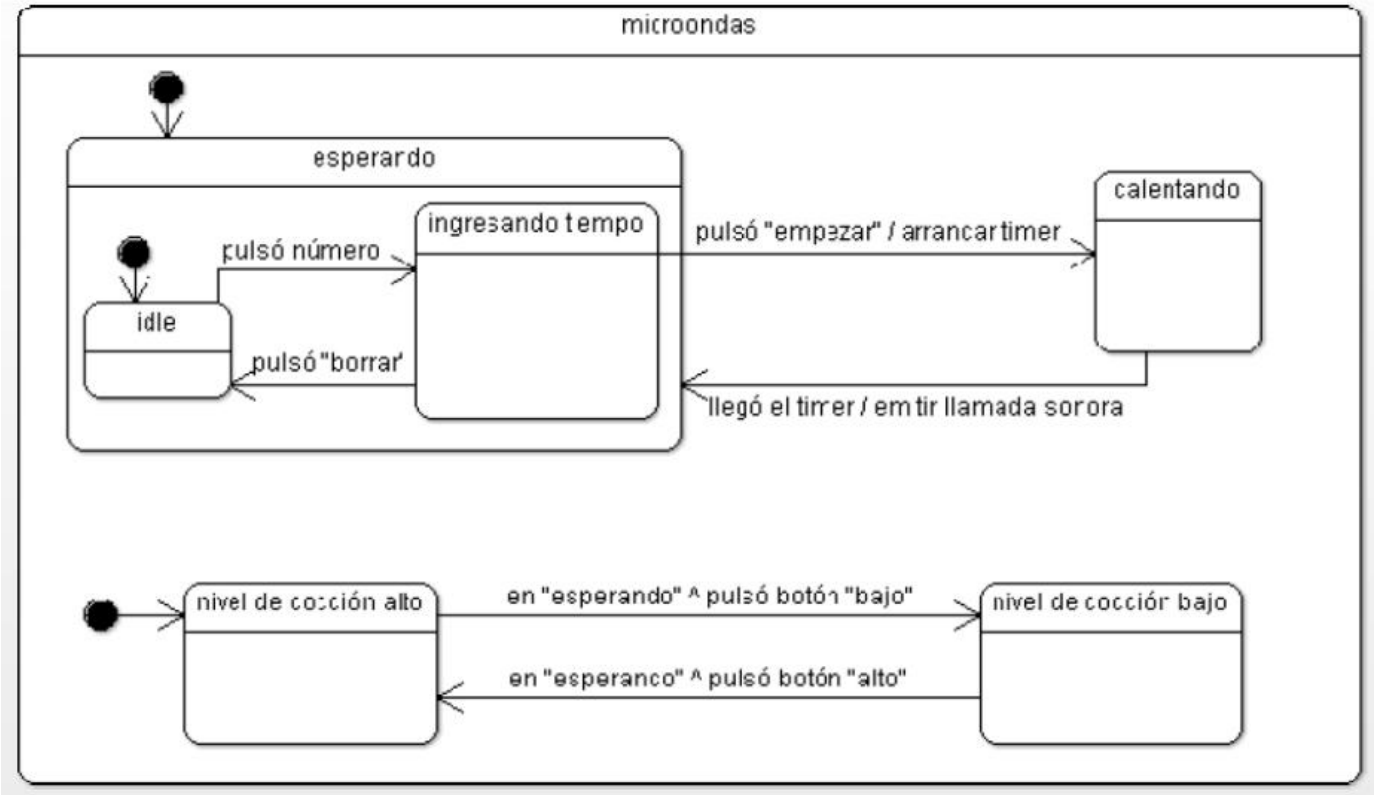
- ***Tareas (do):*** Acciones que se realizan mientras se permanezca en el estado.
- ***Acción de Entrada:*** Se ejecuta únicamente cuando se entra en el estado, independientemente de la transición que lo genera.
- ***Acción de Salida:*** Se ejecuta únicamente cuando se sale dell estado, independientemente de la transición que lo genera.
- ***Acción en Transición:*** Se ejecuta únicamente en una transición entre estados.

Estados. Condiciones de guarda

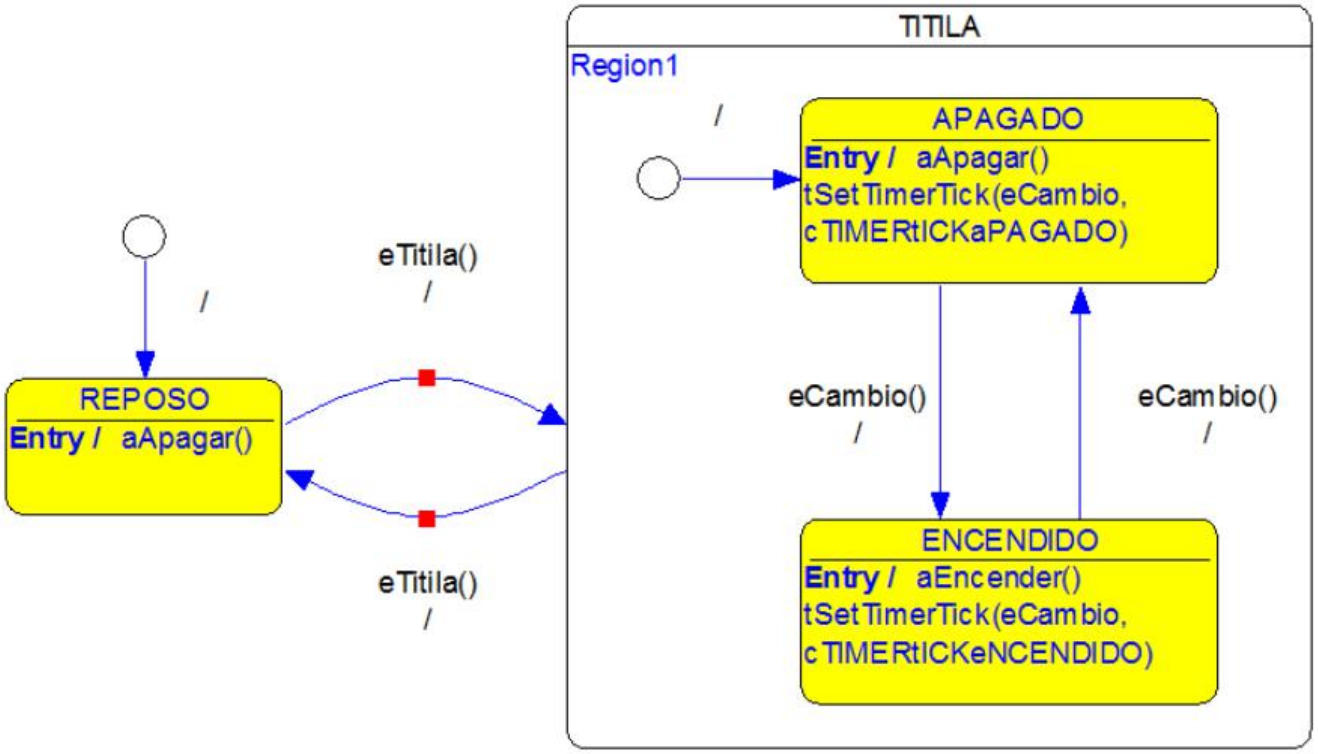


- Las condiciones de guarda son expresiones booleanas que permiten habilitar o deshabilitar acciones o transiciones en función de su evaluación.
- Son evaluadas dinámicamente.
- Tienen un fuerte impacto en la generación de código.

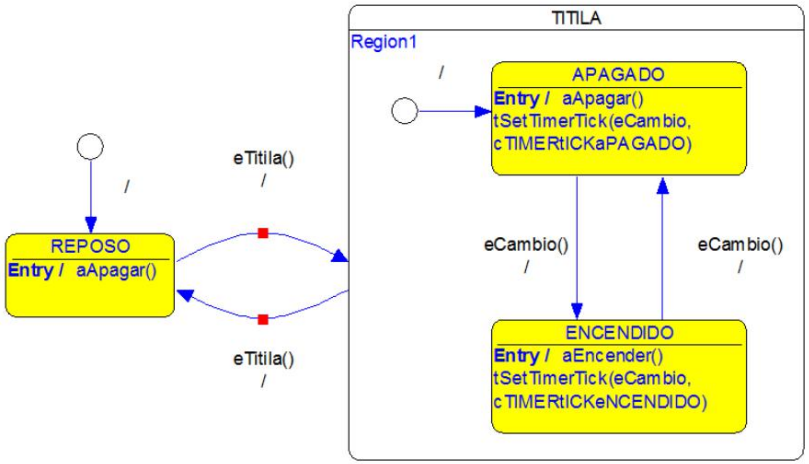
Concurrencia



Estados jerárquicos



Transiciones Internas

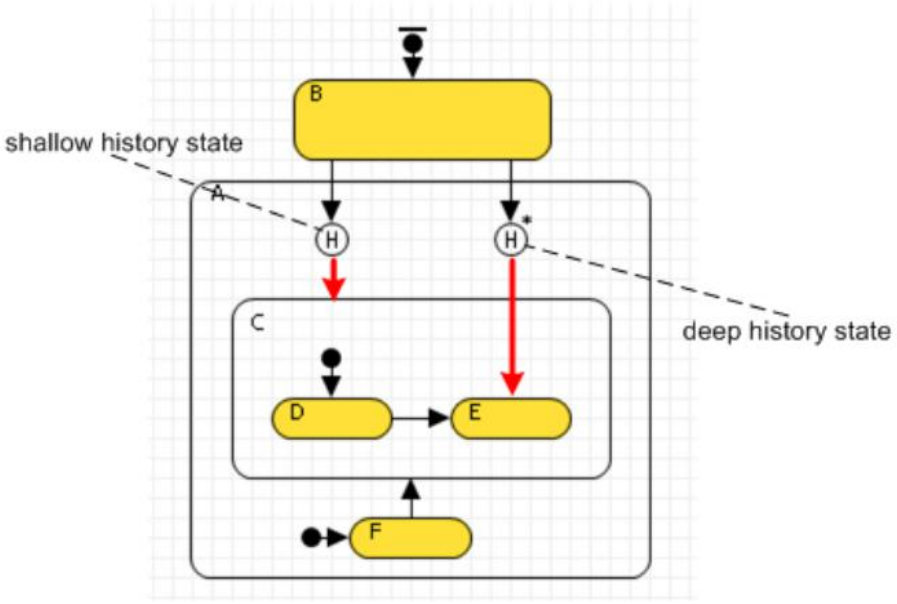


- Las transiciones generadas por los eventos eCambio() no van a generar acciones Entry o Exit en el estado TITILA.
- Estas transiciones se las conoce como transiciones internas.

Pseudoestados

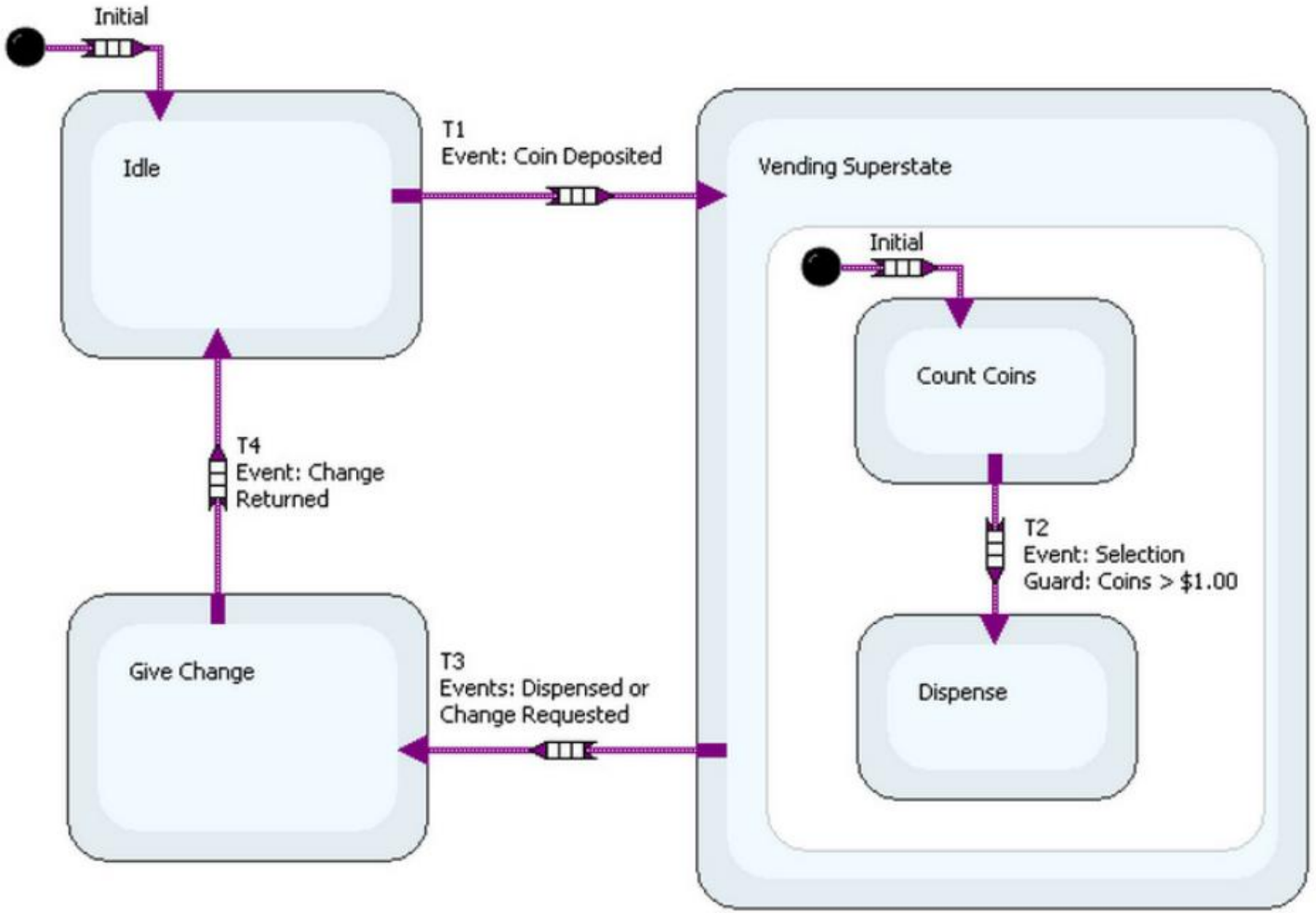
- Son similares a los estados, con la diferencia que el sistema no puede permanecer en ellos. Tipos:
  - ***Inicial (Reset).*** Se indica con un círculo vacío.
  - ***Junturas.*** Se indican con un punto negro. Sirven para juntar o bifurcar transiciones que tienen partes en común.
  - ***Historia.*** 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.*** Con un fork se “abre” una transición para que pueda terminar en estados de máquinas concurrentes. Join hace lo contrario.
  - ***Final.*** Se indica con dos círculos concéntricos, el interior negro y el exterior vacío.

Pseudoestados. Estados Historia.

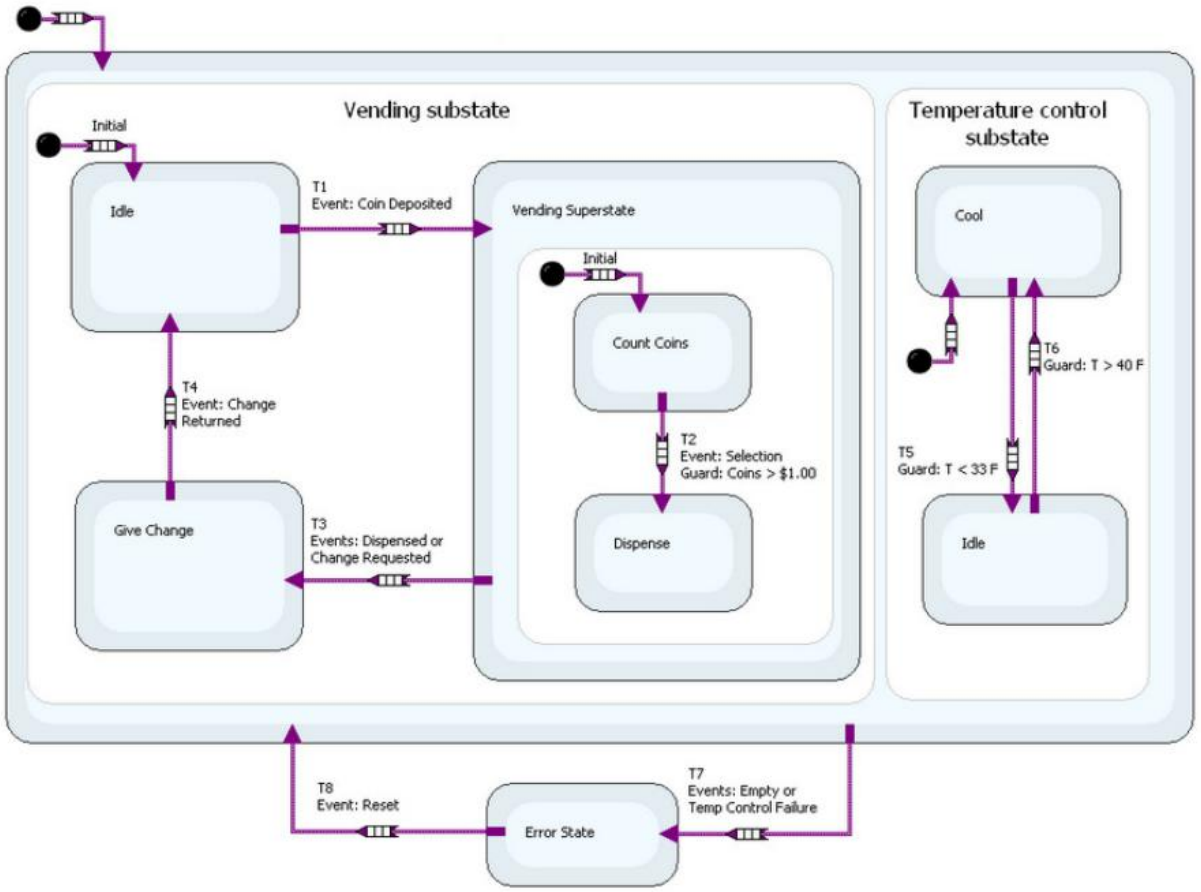


- El estado historia, recuerda al estado de la jerarquía actual.
- El estado historia profunda, recuerda el estado más internode la jerarquía.

Ejemplo. Expendedor de bebidas.



Ejemplo. Expendedor de bebidas.



Cómo construir un Statechart.

1. Identificar eventos.
2. Identificar acciones.
3. Identificar estados.
4. Agrupar por jerarquías
5. Agrupar por concurrencia
6. Agregar las transiciones y acciones.

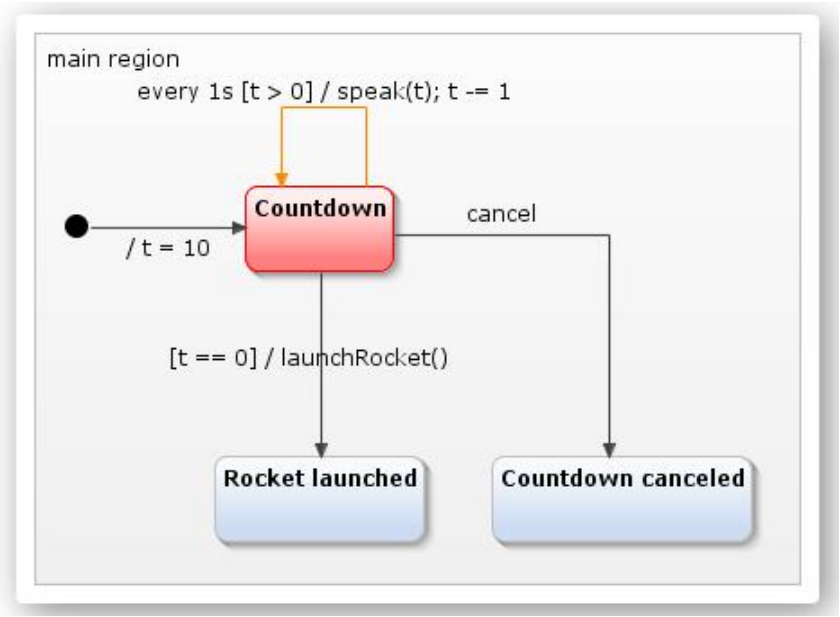
Herramientas

- Statemate and Rhapsody(I−Logix, http://www.ilogix.com)
- Rational Suite Development Studio Real−Time (Rational
- Software Corp., http://www.rational.com)
- BetterState (WindRiver Systems, http://www.wrs.com)
- Stateflow (MathWorks, http://www.mathworks.com)
- VisualState (IAR, http://www.iar.com)
- ObjectGeode (Telelogic, http://www.telelogic.com)
- Yakindu (https://www.itemis.com/en/yakindu/state-machine/)

Capacidades Esperadas.

- Generación del statechart.
- Simulación de los statecharts
- Generación automática de código.

Herramientas.



- Se utilizará Yakindu Community Edition.
  - Integrada/Integrable a Eclipse.
  - Sin costo (Community)
  - Genera y simula statecharts.
  - Generación de código en C.

Bibliografía.

- Object-Oriented Analysis and Design with Applications (3rd Edition). Grady Booch.
- The Unified Modeling Language Reference Manual. Booch, Jacobson, Rumbaugh.
- Object-Oriented Analysis and Design with Applications (3rd Edition). Grady Booch.
- The Unified Modeling Language Reference Manual. Booch, Jacobson, Rumbaugh.
- Practical Statecharts in C/C++: Quantum Programming for Embedded Systems. Miro
- Samek.
- Statecharts. Andrés Djordjalian. Seminario de Sistemas Embebidos. FIUBA.
- www.indicart.com.ar/seminario-embebidos/Statecharts.pdf
- Diagrama de Estado. Juan Manuel Cruz.
- http://laboratorios.fi.uba.ar/lse/seminario/material-
- 1erC2011/Sistemas%20Embebido%202011%201er%20Cuatrimestre%20-
- %20Diagrama%20de%20Estado%20-%20Cruz.pdf