



Ingeniería de Software I

Clase 4 - Diagrama de transición de estados - DTE

Técnicas de Especificación de Requerimientos

Estáticas

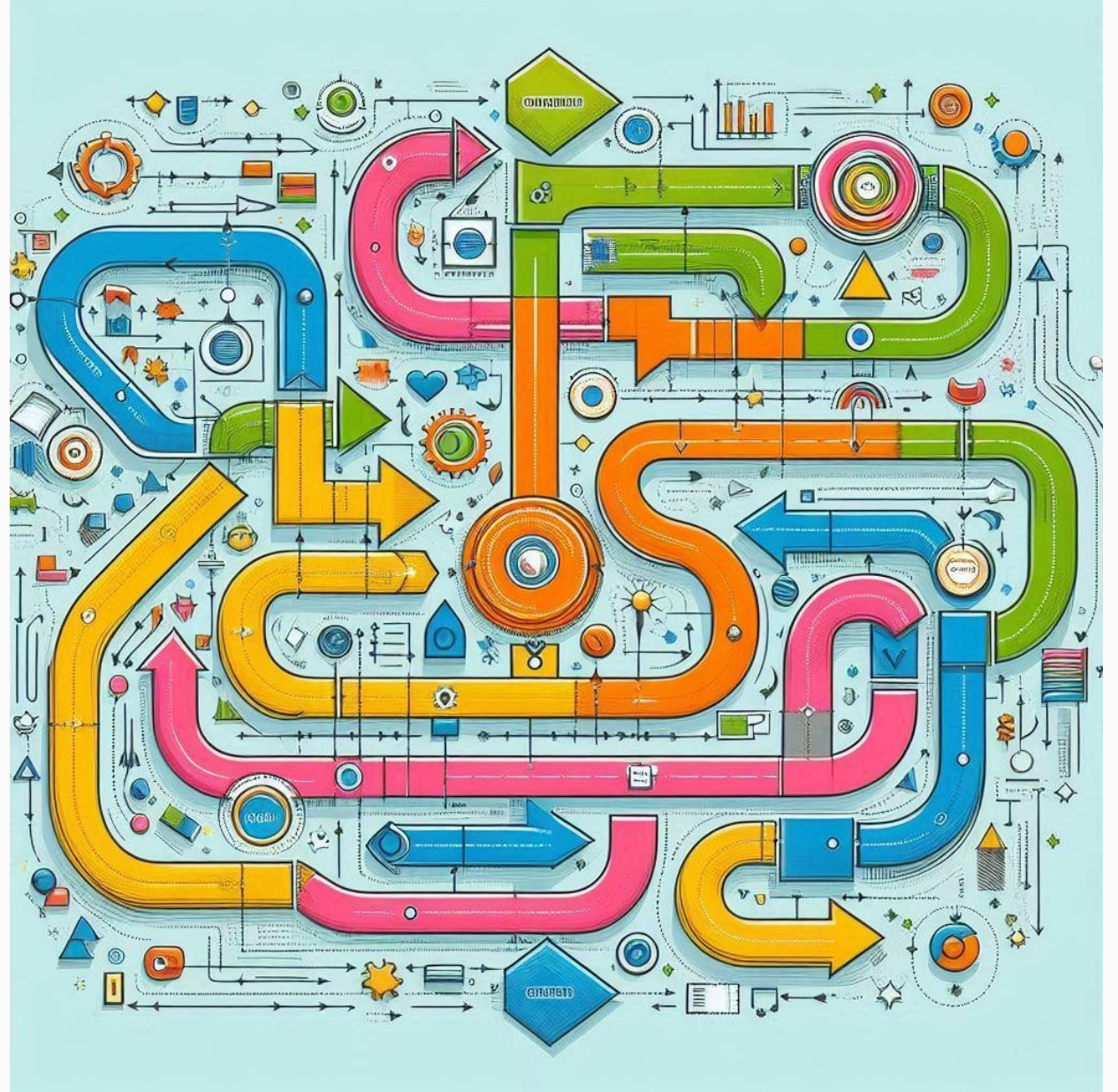
- Se describe el sistema a través de las *entidades u objetos, sus atributos y sus relaciones con otros. No describe como las relaciones cambian con el tiempo.*
- Cuando el tiempo no es un factor mayor en la operación del sistema, es una descripción útil y adecuada.
- Ejemplos: Referencia indirecta, Relaciones de recurrencia, Definición axiomática, Expresiones regulares, Abstracciones de datos, entre otras.

Técnicas de Especificación de Requerimientos

Dinámicas

- Se considera un sistema en función de los cambios que ocurren a lo largo del tiempo.
- Se considera que el sistema está en un estado particular hasta que un estímulo lo obliga a cambiar su estado.
- Ejemplos: Tablas de decisión, Diagramas de transición de estados, Tablas de transición de estados, Diagramas de persianas, Diagramas de transición extendidos, Redes de Petri, entre otras.

DTE



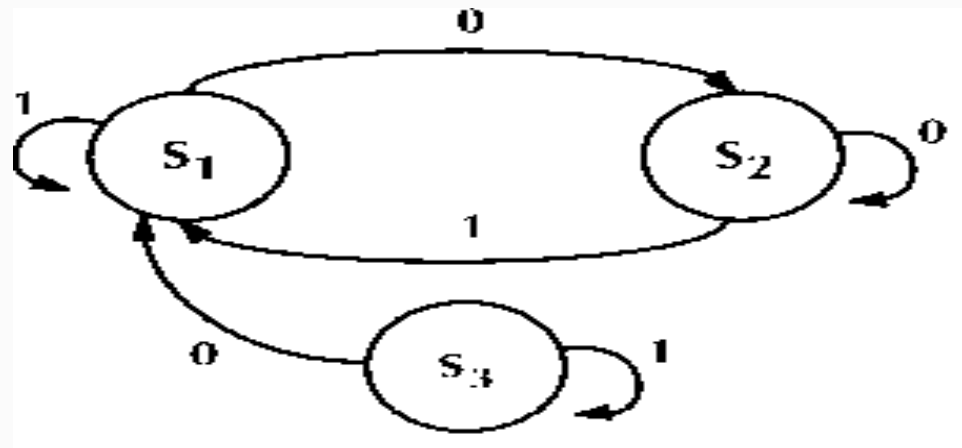
Técnicas de Especificación de Requerimientos Dinámicos

Máquinas de Estado Finito

- Describe al sistema como un conjunto de estados donde el sistema reacciona a ciertos eventos posibles (externos o internos).

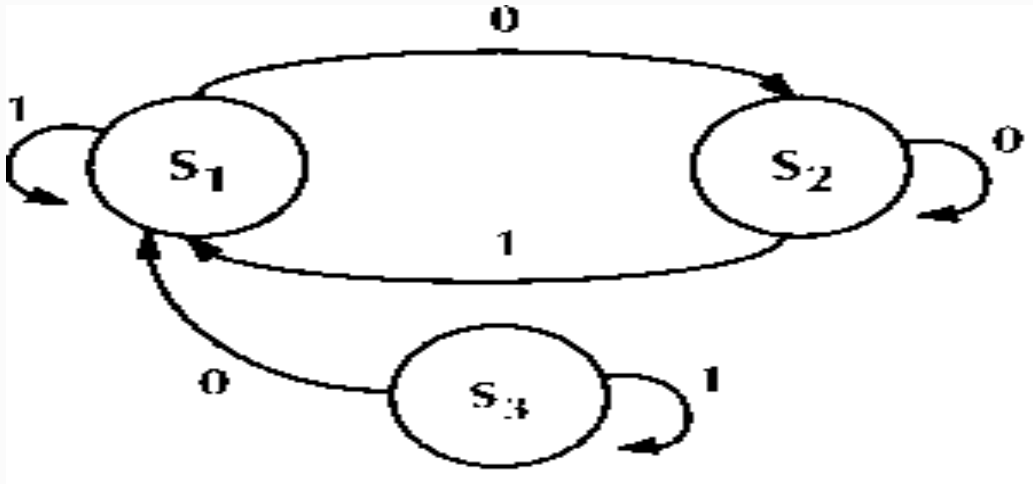
$$f(S_i, C_j) = S_k$$

- Al estar en el estado S_i , la ocurrencia de la condición C_j hace que el sistema cambie al estado S_k .



Técnicas de Especificación de Requerimientos Dinámicos

Máquinas de Estado Finito



$$f(S1, 0) = S2$$

$$f(S1, 1) = S1$$

$$f(S2, 0) = S2$$

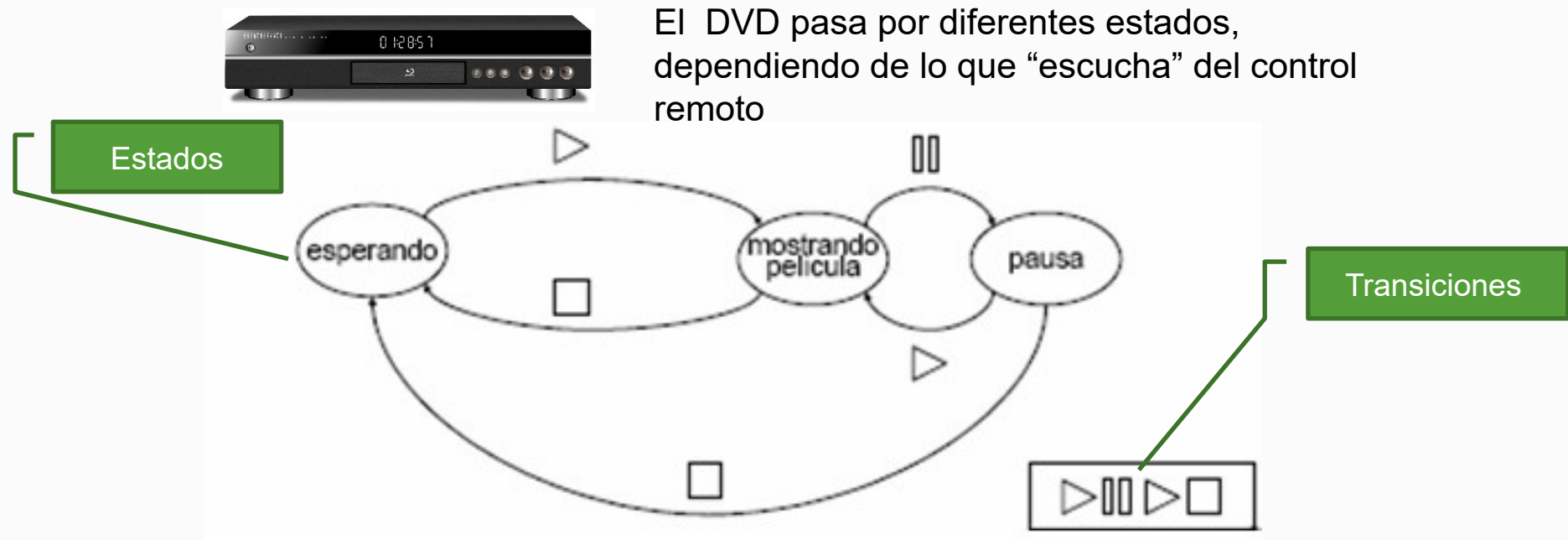
$$f(S2, 1) = S1$$

$$f(S3, 0) = S1$$

$$f(S3, 1) = S3$$

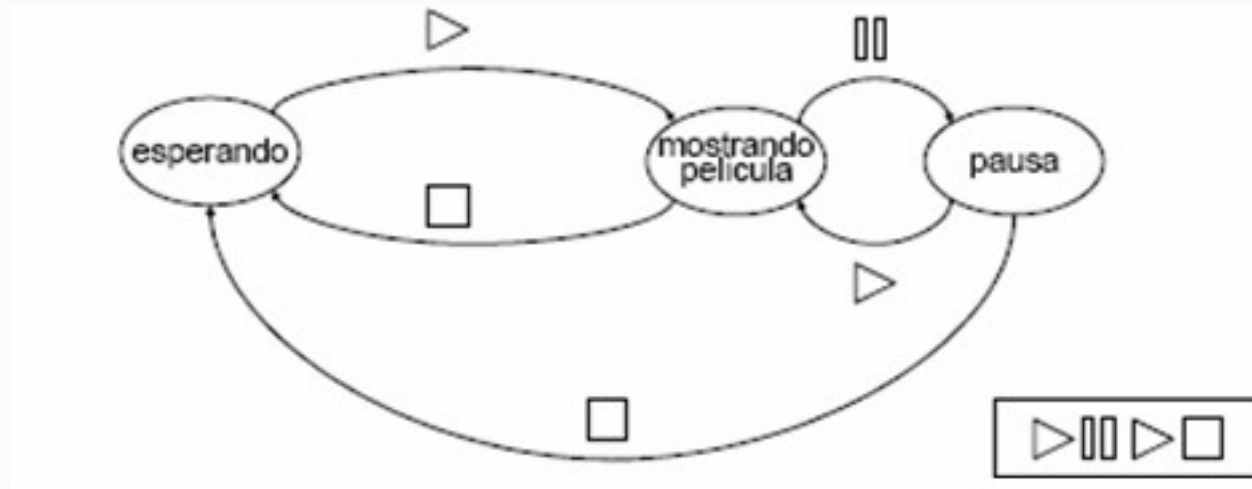
Técnicas de Especificación de Requerimientos Dinámicas

Máquinas de Estado Finito



Técnicas de Especificación de Requerimientos Dinámicos

Máquinas de Estado Finito



$f(A, \triangleright) = B$

$f(B, \square) = A$

$f(B, \parallel) = C$

$f(C, \triangleright) = B$

$f(C, \square) = A$

A: esperando

B: mostrando

C: pausa

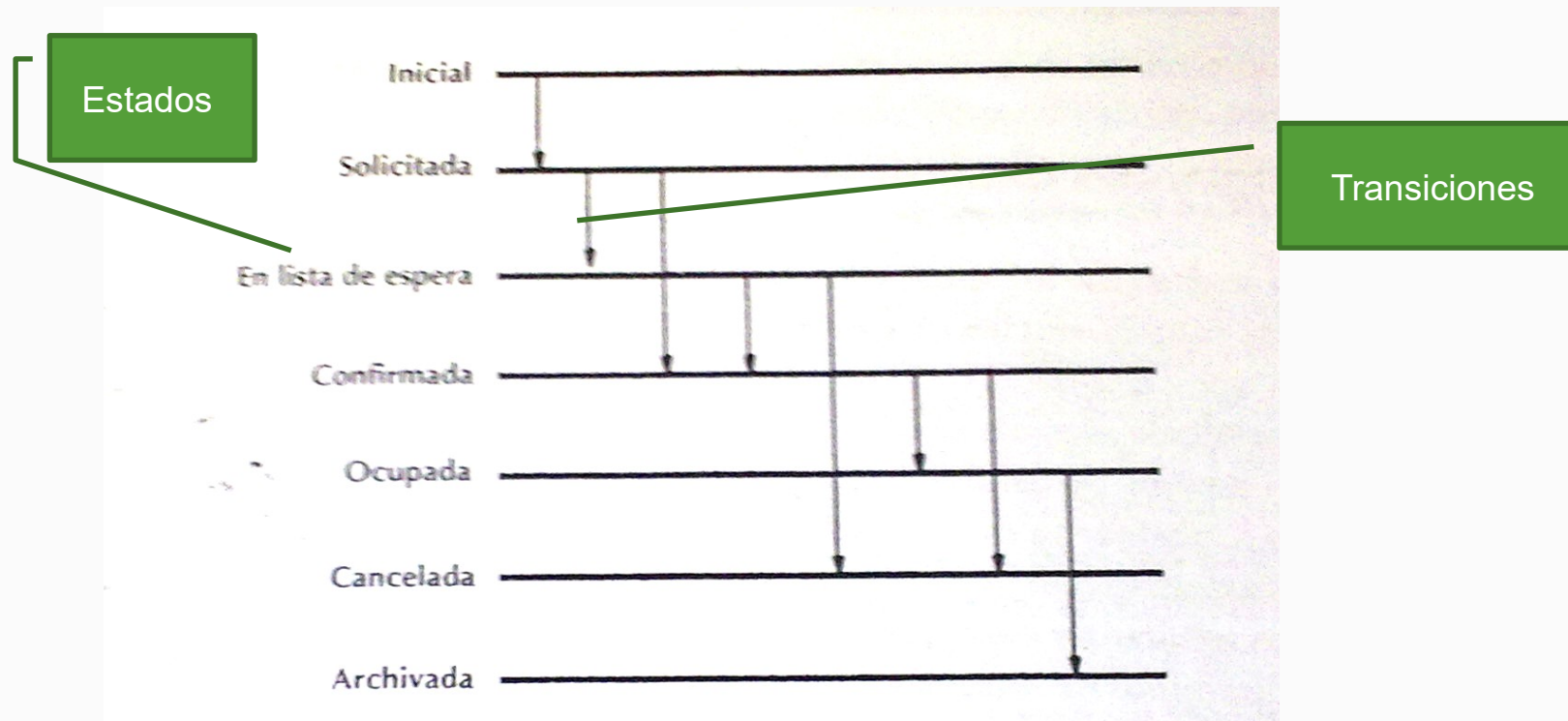
Técnicas de Especificación de Requerimientos Dinámicas

Máquinas de Estado Finito

- Definición formal
 - Formalmente, un autómata finito (AF) puede ser descrito como una 5-tupla (S, Σ, T, s, A) donde:
 - Σ es un alfabeto;
 - S un conjunto de estados;
 - T es la función de transición;
 - s es el estado inicial;
 - A es un conjunto de estados de aceptación o finales.

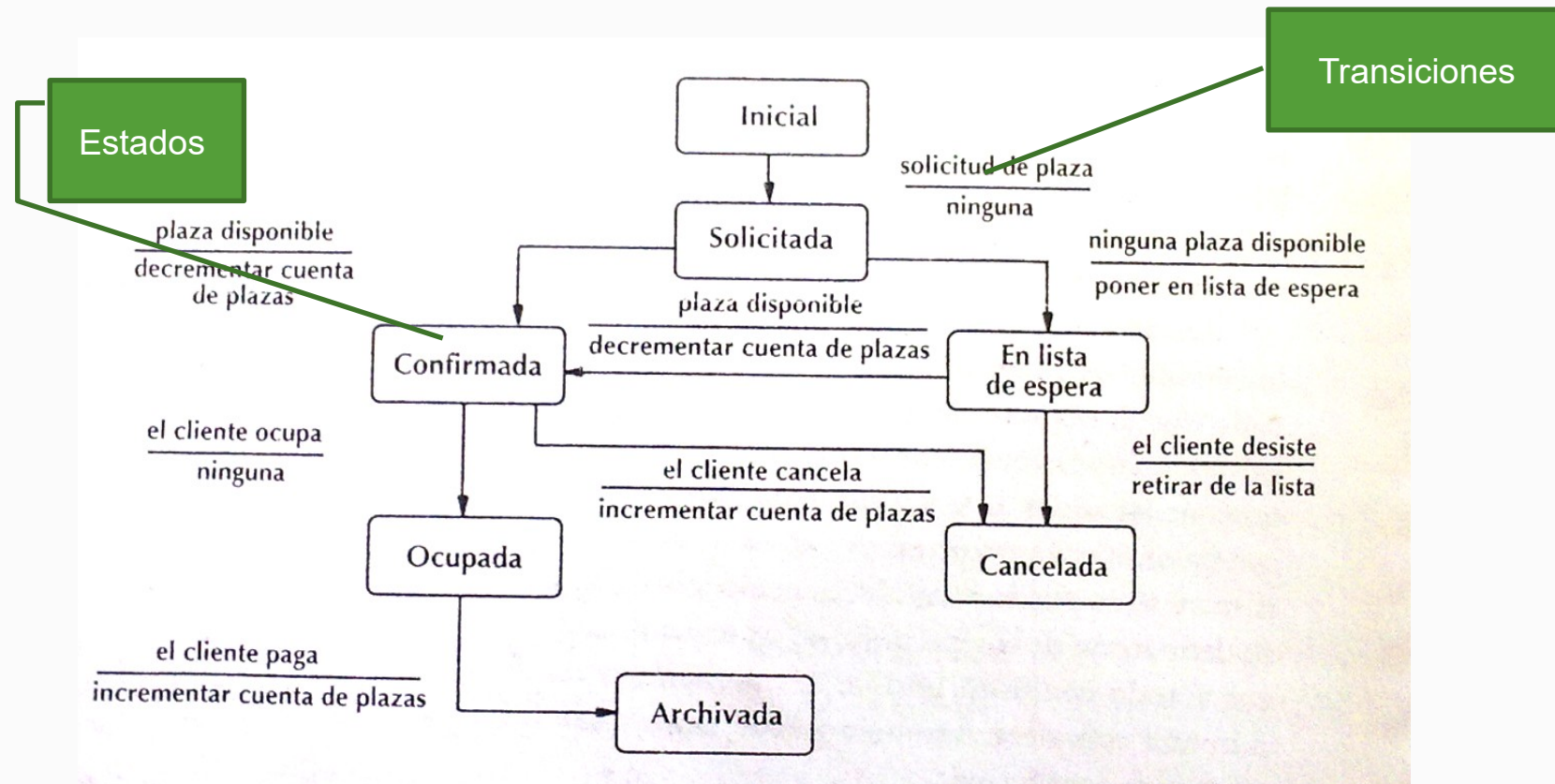
Técnicas de Especificación de Requerimientos Dinámicos

Representación en gráfico de persiana



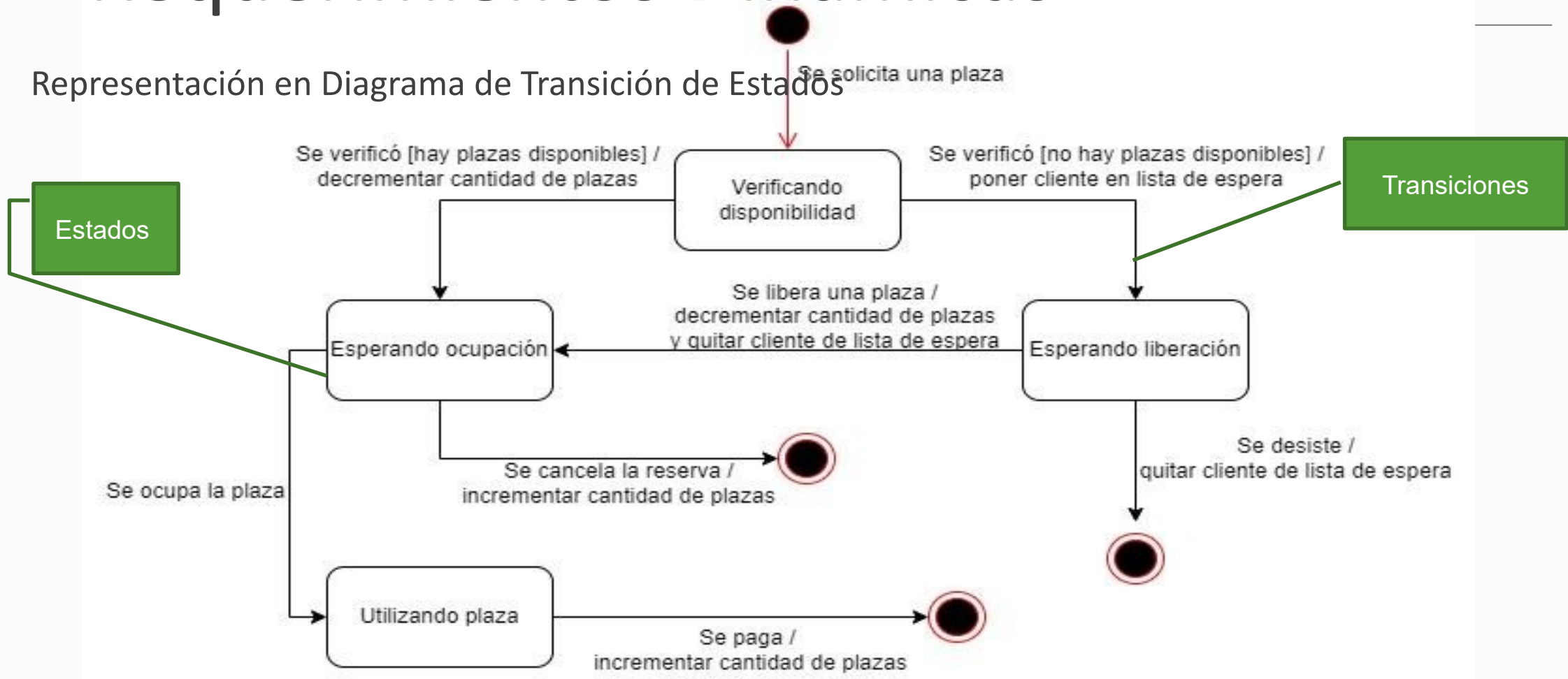
Técnicas de Especificación de Requerimientos Dinámicos

Representación en máquina de estado finito



Técnicas de Especificación de Requerimientos Dinámicos

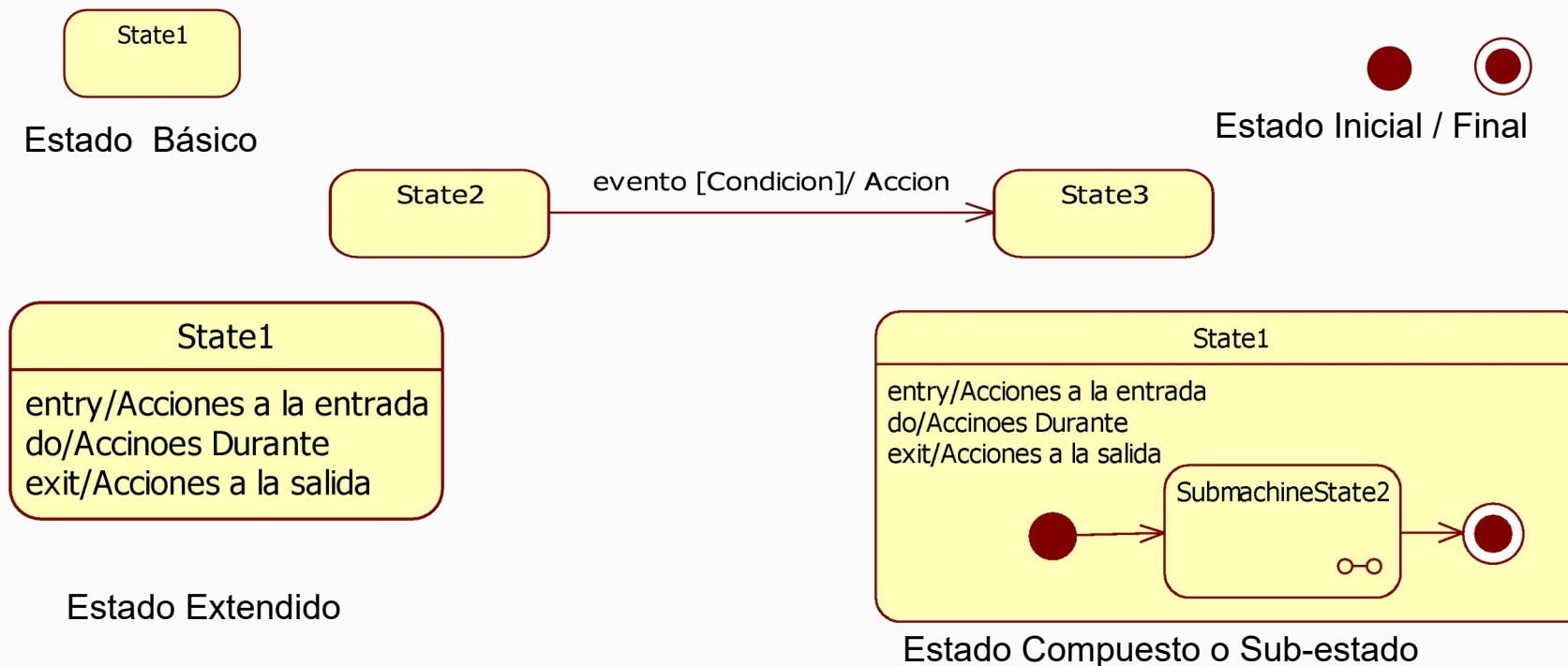
Representación en Diagrama de Transición de Estados



Técnicas de Especificación de Requerimientos Dinámicos

Máquinas de Estado Finito

- Notación UML Diagrama de Transición de Estado (DTE)



Evento

Es un **suceso significativo** que debe tenerse en cuenta, que influye en el comportamiento y evolución del sistema.

Tiene lugar en un punto del tiempo y carece de duración respecto a la granularidad temporal del sistema.

No tiene sentido preguntarse por lo que sucede mientras está teniendo lugar el evento.

Transición

Las transiciones se producen como consecuencia de eventos. Pueden o no tener un procesamiento asociado



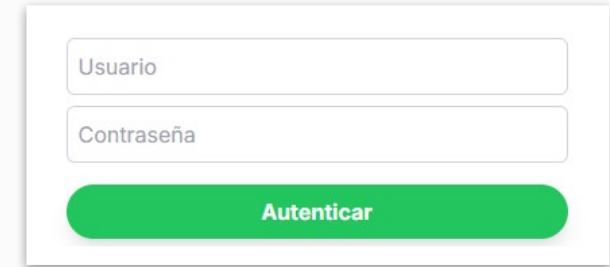
- Evento: obligatorio
- Condición: opcional, depende del problema, puede haber transiciones sin condiciones
- Acción: opcional, puede haber transiciones sin acciones

Técnicas de Especificación de Requerimientos Dinámicas

Construcción de un DTE

- 1- Identificar los estados
- 2- Si hay un estado complejo se puede explotar
- 3- Identificar el estado inicial
 - Desde el estado inicial, se identifican los cambios de estado con flechas
- 4- Se analizan las condiciones y las acciones para pasar de un estado a otro
- 5- Se verifica la consistencia:
 - Se han definido todos los estados
 - Se pueden alcanzar todos los estados
 - Se pueden salir de todos los estados
 - En cada estado, el sistema responde a todas las condiciones posibles (normales y anormales)

Técnicas de Especificación de Requerimientos Ejercicio DTE



Mockup of a login form:

- Input field: Usuario
- Input field: Contraseña
- Button: Autenticar

Se requiere modelar el comportamiento de un sistema de autenticación simple para una aplicación móvil. El sistema debe seguir las siguientes reglas:

Cuando se abre la aplicación por primera vez, el sistema muestra una "Pantalla de Inicio". Desde allí el usuario puede presionar el botón "Ingresar". Esto lo lleva al "Formulario de Ingreso".

Proceso de autenticación: en el "Formulario de Ingreso", si el usuario ingresa las credenciales correctas y presiona "Autenticar", el sistema activa la sesión del usuario. Si las credenciales son incorrectas, el sistema muestra un mensaje de error y permanece en el "Formulario de Ingreso".

Una vez iniciada la sesión, el usuario puede presionar el botón "Cerrar Sesión", lo que lo devuelve a la "Pantalla de Inicio".

Inactividad: después de 5 minutos de inactividad, la sesión del usuario expira automáticamente, y el sistema regresa a la "Pantalla de Inicio".

Si la aplicación se cierra, el sistema finaliza, independientemente del estado en el que se encuentre.

Técnicas de Especificación de Requerimientos Ejercicio DTE

Funciones

- Inicialmente (al abrir la app) visualiza la pantalla inicial
- Visualizar formulario de ingreso
- Ingresar datos de autenticación (correctos o incorrectos)
- Presionar botón cerrar sesión
- Dejar la aplicación inactiva X tiempo
- Finaliza si la aplicación se cierra

Técnicas de Especificación de Requerimientos Ejercicio DTE

Debemos:

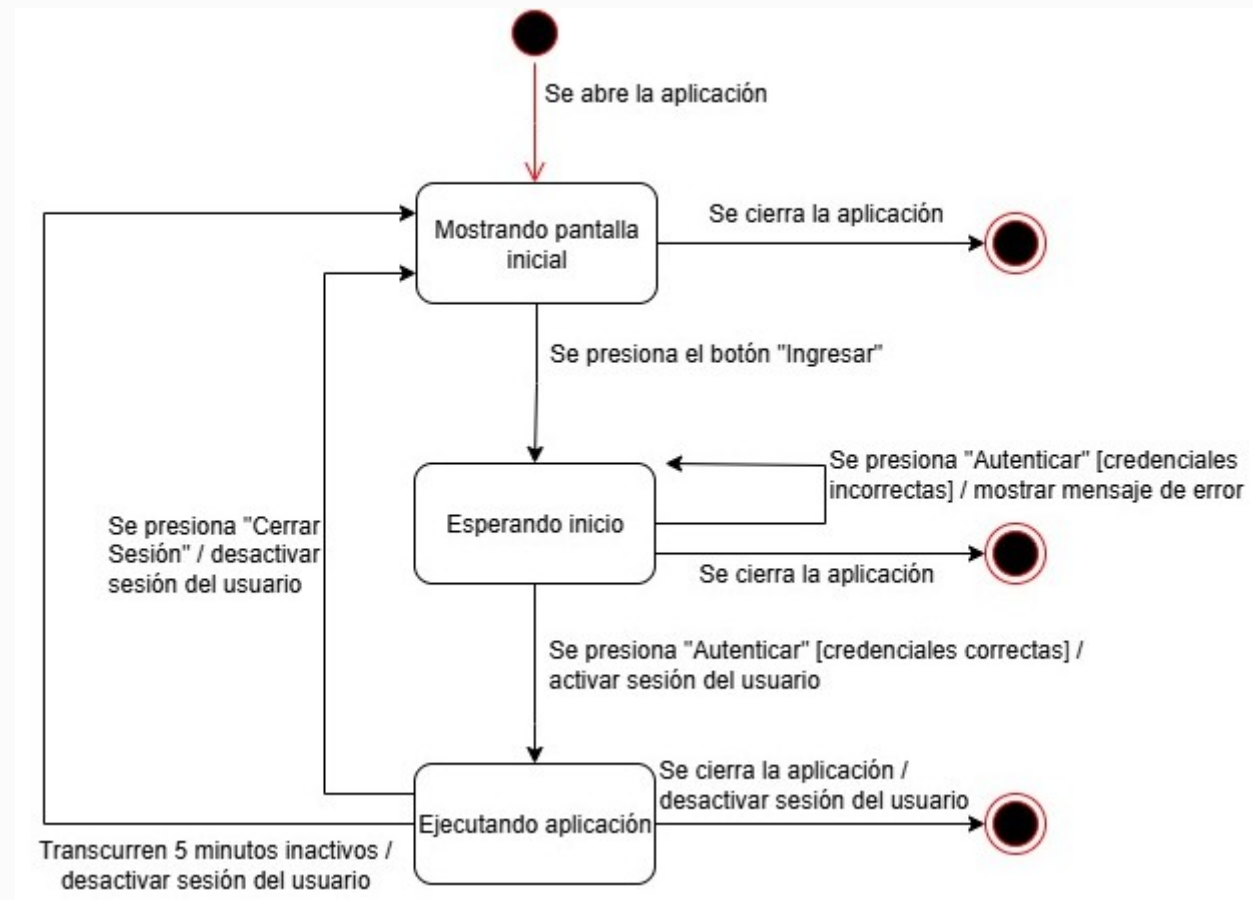
- 1- Identificar los estados
- 2- Si hay un estado complejo, explotarlo
- 3- Identificar el estado inicial
- 4- Analizar las condiciones y las acciones para pasar de un estado a otro
- 5- Verificar la consistencia

Técnicas de Especificación de Requerimientos Ejercicio DTE

Veámoslo en funcionamiento:



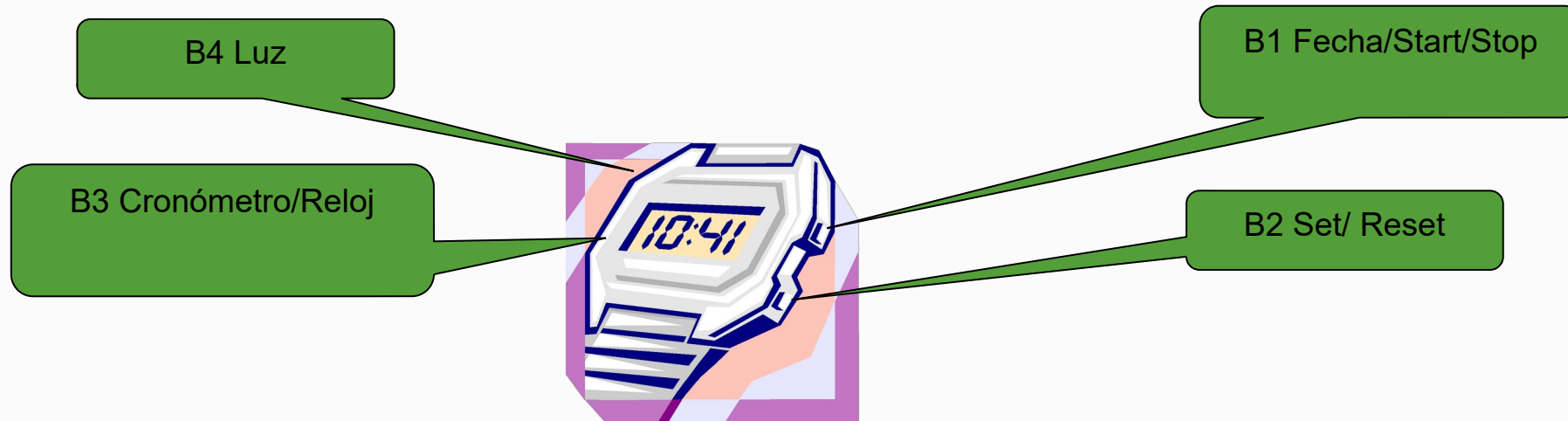
Técnicas de Especificación de Requerimientos Ejercicio DTE



Técnicas de Especificación de Requerimientos Ejercicio DTE

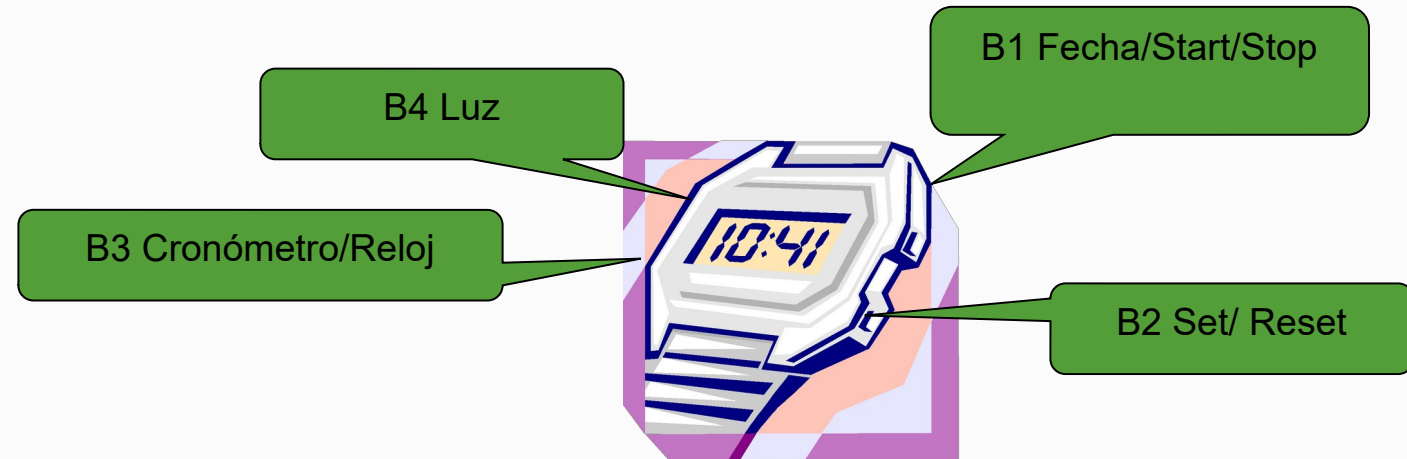
Reloj Cronómetro

- El reloj posee una pantalla y 4 botones



Técnicas de Especificación de Requerimientos Ejercicio DTE

Reloj Cronómetro



Veámoslo en funcionamiento: **Simulación**

Técnicas de Especificación de Requerimientos Ejercicio DTE

Funciones

- Inicialmente (al colocar la pila) visualiza la hora prefijada
- Visualizar la hora
- Visualizar la fecha
- Modificar Hora y Fecha
- Encender la Luz por 5 seg.
- Iniciar / Detener / Resetear Cronómetro
- Deja de funcionar al finalizarse la pila

Técnicas de Especificación de Requerimientos Ejercicio DTE

1- Identificar los estados

- Visualizando hora
- Visualizando fecha
- Visualizando funciones cronometro
- Cronometrando
- Configurando hora y fecha

2- Identificar estados complejos

- No es necesario

3- Estado inicial

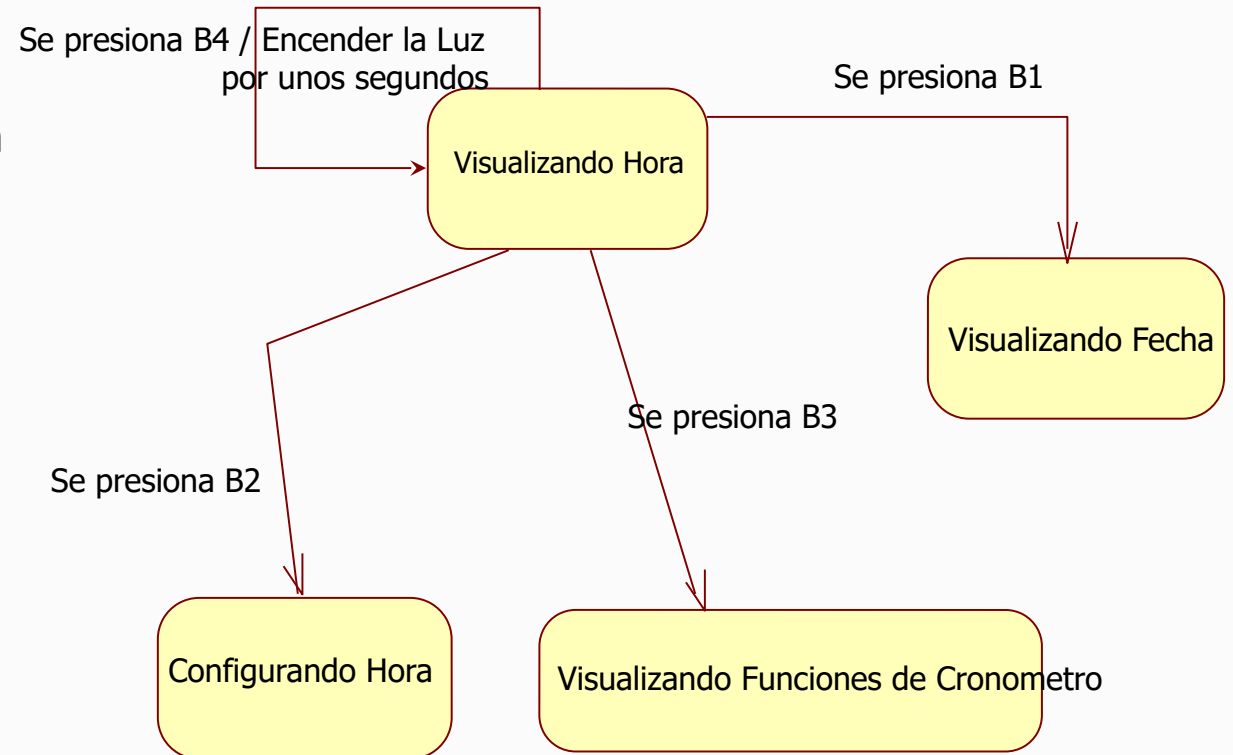
- En este caso, el sistema inicia al colocarse la pila y pasaría al estado visualizando hora



Técnicas de Especificación de Requerimientos Ejercicio DTE

4- Visualizando hora

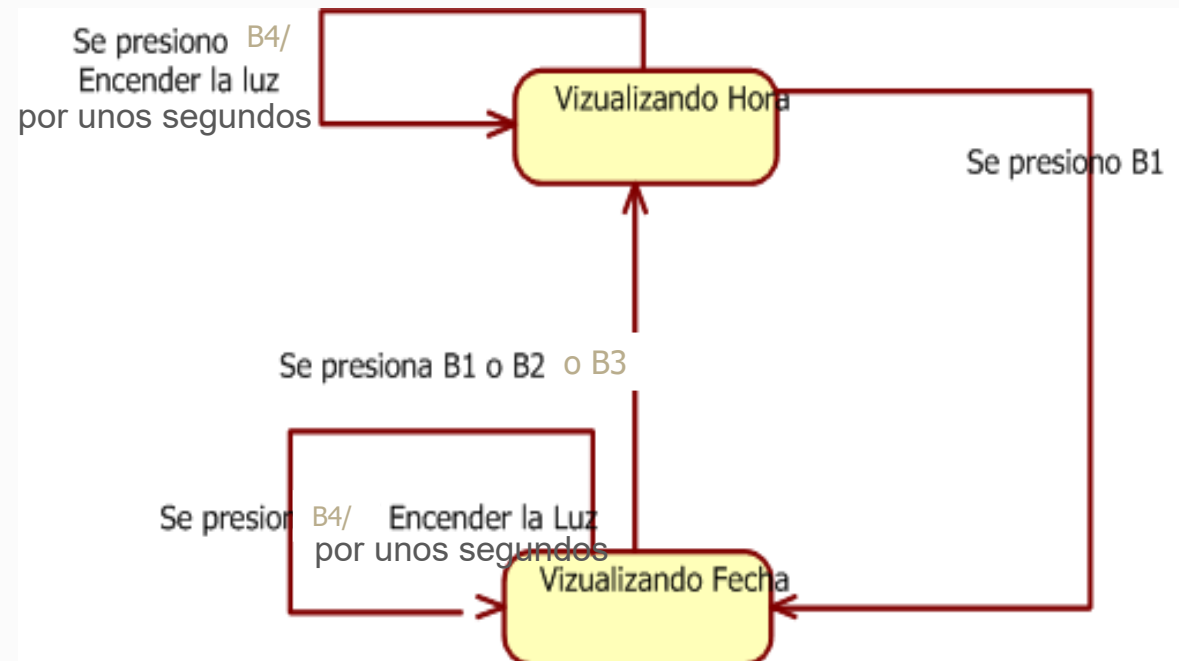
- Se presiona B1 Visualiza la fecha
- Se presiona B2 Modificar la hora y fecha
- Se presiona B3 Visualiza el cronometro
- Se presiona B4 Enciende la luz



Técnicas de Especificación de Requerimientos Ejercicio DTE

4- Visualizando fecha

- Estando en el estado Visualizando fecha , presionando B1 o B2 o B3 vuelve a visualizar la hora
- En Cualquier Momento se puede encender la luz con el botón B4

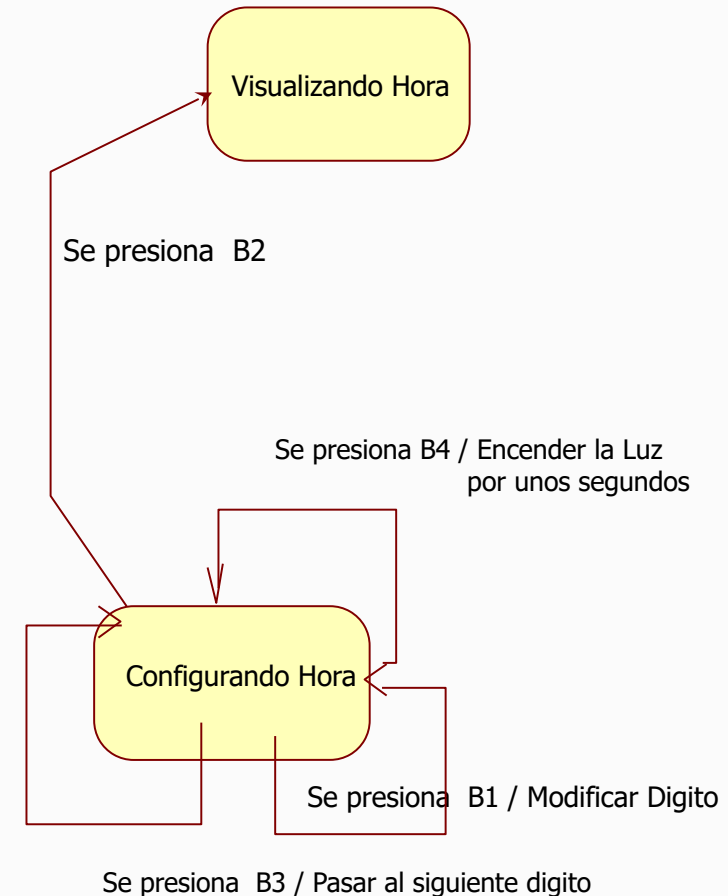


Técnicas de Especificación de Requerimientos Ejercicio DTE

4- Configurando Hora y Fecha

- Se presiona B1 modifico el digito
- Se presiona B2 vuelve a visualizar la hora
- Se presiona B3 modifico el digito a modificar
 - Hora, minuto, segundo, día, mes
- Se presiona B4 enciende la luz

4- Continuar con todos los estados

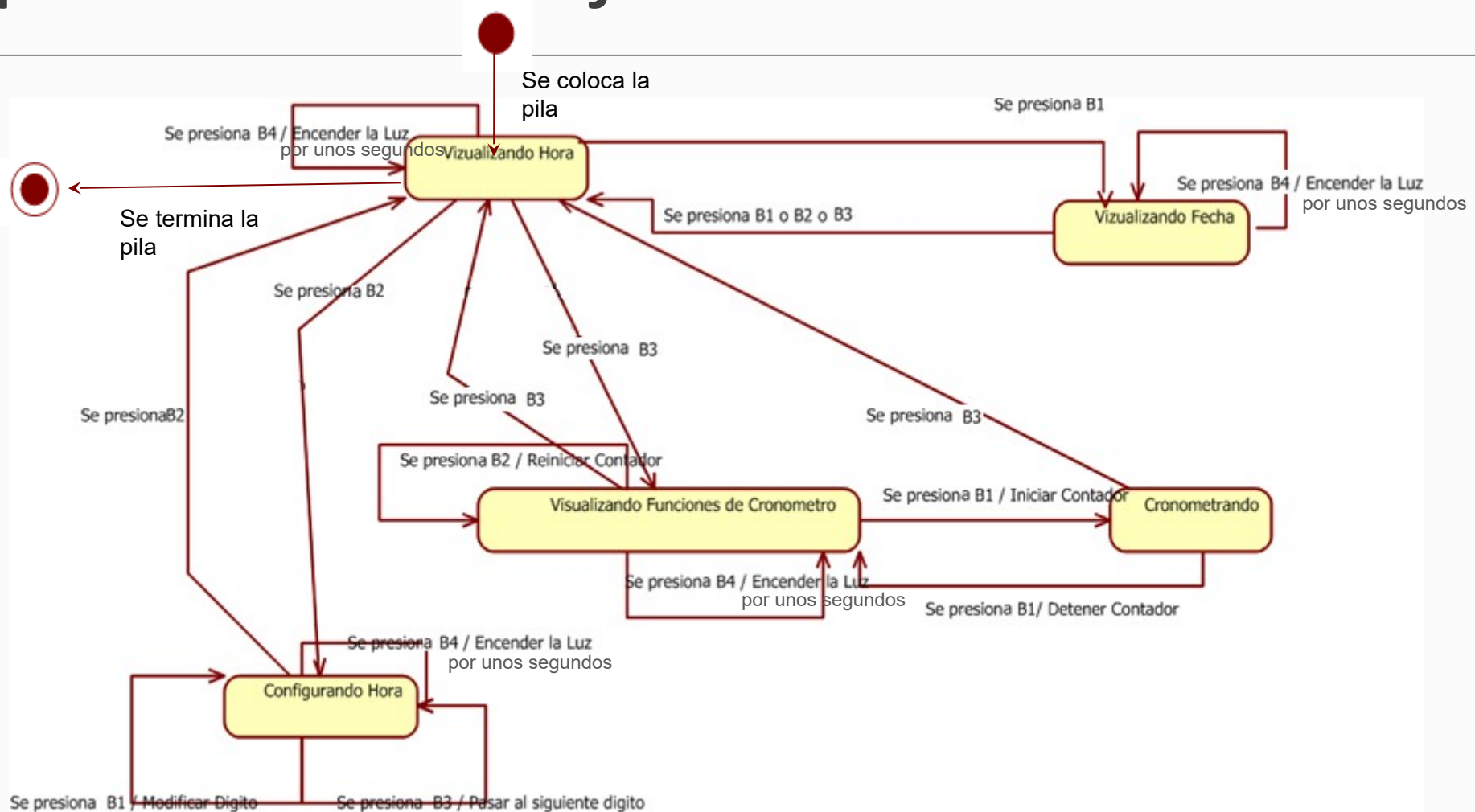


Técnicas de Especificación de Requerimientos Ejercicio DTE

5- Se verifica la consistencia:

- Se han definido todos los estados
- Se pueden alcanzar todos los estados
- Se pueden salir de todos los estados
- En cada estado, el sistema responde a todas las condiciones posibles (normales y anormales)

Técnicas de Especificación de Requerimientos Ejercicio DTE



Ventajas del DTE

- ❖ **Mejora la comprensión:** ofrece un mapa visual que hace fácil entender cómo un sistema reacciona a diferentes eventos.
- ❖ **Previene errores:** ayuda a encontrar fallos de lógica y caminos que no deberían existir, todo esto en la etapa de diseño, lo que ahorra tiempo y esfuerzo en la codificación.
- ❖ **Valida el diseño:** confirma que todos los requisitos del sistema han sido considerados y que el comportamiento es coherente.
- ❖ **Facilita las pruebas:** proporciona un plan para crear pruebas, asegurando que se validan todas las interacciones posibles del sistema.

Inconvenientes del DTE

- ❖ **Complejidad:** pueden volverse muy complejos y difíciles de manejar en sistemas con una gran cantidad de estados y transiciones
- ❖ **Enfoque en un solo objeto:** un DTE típicamente modela el comportamiento de un único objeto o componente. Para entender el sistema completo, se necesita una colección de DTEs
- ❖ **No modela la concurrencia:** los DTEs básicos no son ideales para modelar sistemas concurrentes, donde varios estados pueden cambiar al mismo tiempo.

¿Cuándo usar DTE?

- ❖ **Sistemas de tiempo real:** cuando el software está impulsado por eventos externos, como un sistema de control para una máquina o un dispositivo.
- ❖ **Modelado del comportamiento de la interfaz de usuario:** para aplicaciones con interfaces de usuario complejas, un DTE puede mostrar cómo la pantalla o un componente de la interfaz cambia en respuesta a las acciones del usuario.
- ❖ **Sistemas con un número finito de estados:** cuando el componente de software puede estar en un número limitado de estados bien definidos (por ejemplo, una válvula que puede estar "abierta" o "cerrada").
- ❖ **Diseño de algoritmos de control:** en el diseño de software para sistemas de control, donde el estado actual del sistema determina su próximo comportamiento.

Bibliografía

Libros Utilizados en la Teoría

- Pfleeger, Capítulo 4 , Ingeniería de Software, Pearson-Prentice Hall 2002
- Sommerville Ian, Capítulo 5, Ingeniería de Software, Addison-Wesley 2011