

# Programación concurrente

Master de Computación

*I Conceptos y recursos para la programación concurrente:*

**I.1 Motivación de la programación concurrente.**



**J.M. Drake**

**M. Aldea**

# Motivación de la programación concurrente

---

- La crisis del software.
- Programas secuenciales, concurrentes y de tiempo real.
- Programación concurrente.
- Programación de tiempo real.
- Entornos hardware para programación concurrente.
- Ejemplo de un problema concurrente.

## LA CRISIS DEL SOFTWARE.

---

- Conjunto de tópicos relacionados con la problemática asociada al desarrollo de software:  
**"Construir una aplicación software es una tarea mucho más compleja de lo que parece al iniciarla"**
- Aspectos de esta problemática son:
  - **Responsiveness:** No satisfacen las expectativas del usuario.
  - **Reliability:** Presentan fallos y su depuración es muy difícil.
  - **Cost:** El costo es difícil de evaluar y mas alto de lo esperado.
  - **Modificability:** Son productos muy rígidos y difíciles de mantener.
  - **Timeless:** Requieren para su ejecución mas tiempo del previsto.
  - **Transportability:** Hay problemas para migrar entre plataforma.
  - **Efficiency:** Sólo utilizan una parte de la capacidad de hardware.

# CAUSAS DE LA CRISIS DEL SOFTWARE.

---

- Causas profundas de la crisis del software son:
  - La metodología en cascada que linealiza el proceso de desarrollo.
  - La metodología de modularización estructurada hace que el software sea inflexible y difícil de mantener.
  - Los programadores no tienen formación en ingeniería software.
  - La “inercia” de las empresas e instituciones dificulta la introducción de innovaciones.
  - La estructura secuencial de Von Newman no se adapta a los problemas que se abordan.

# PROGRAMAS Y LÍNEAS DE FLUJO DE CONTROL.

---

- Un programa se compone de:
  - Sentencias: Establecen las actividades (operaciones y verificaciones) que ejecuta el sistema.
  - Línea de flujo de control (Thread): Establece el orden en que se ejecutan las sentencias.
  
- Según las características de la línea de flujo de control los programas se clasifican en:
  - Secuenciales.
  - Concurrentes (tipo especial de sistemas concurrentes: Tiempo real).

# PROGRAMAS SECUENCIALES.

---

- Es el estilo de programación que corresponde al modelo conceptual de **Von Neumann**.
- Un programa secuencial tiene una **línea simple** de control de flujo.
- Las operaciones de un programa secuencial están ordenadas de acuerdo con un **orden estricto**.
- El comportamiento de un programa es solo función de las sentencias que lo componen y del orden en que se ejecutan.
- **El tiempo** que tarda en ejecutarse cada operación **no influye** en el resultado de un programa secuencial.
- La **verificación** de un programa secuencial **es sencilla**:
  - Cada sentencia da la respuesta correcta.
  - Las sentencias se ejecutan en el orden adecuado.

# PROGRAMAS CONCURRENTES.

---

- Son programas que tienen múltiples líneas de flujo de control.
- Las sentencias de un programa concurrente se ejecutan de acuerdo con un **orden no estricto**.
- La secuencialización de un programa concurrente es entre **hitos o puntos de sincronización**.
- Un programa concurrente se suele concebir como un **conjunto de procesos** que colaboran y compiten entre sí.
- Para **validar** un programa concurrente:
  - Las operaciones se pueden validar individualmente si las variables no son actualizadas concurrentemente.
  - El resultado debe ser independiente de los tiempos de ejecución de las sentencias.
  - El resultado debe ser independiente de la plataforma en que se ejecuta.

# APLICACIONES DE LOS PROGRAMAS CONCURRENTES

---

- Aplicaciones clásicas:
  - Programación de sistemas multicomputadores.
  - Sistemas operativos.
  - Control y monitorización de sistemas físicos.
- Aplicaciones actuales:
  - Servicios WEB.
  - Sistemas multimedia.
  - Cálculo numérico.
  - Procesamientos entrada/salida.
  - Simulación de sistemas dinámicos.
  - Interacción operador/máquina (GUIs)
  - Tecnologías de componentes.
  - Sistemas embebidos.



# VENTAJAS DE LA PROGRAMACION CONCURRENTE.

---

- Proporciona el **modelo más simple y natural** de concebir muchas aplicaciones.
- Facilita el **diseño orientado a objetos** de las aplicaciones, ya que los objetos reales son concurrentes.
- Hace posible **compartir recursos** y subsistemas complejos.
- En sistemas monoprocesadores **optimiza el uso de los recursos**.
- **Reduce tiempos de ejecución** en plataformas multiprocesadoras.
- Facilita la realización de **programas fiables** replicando componentes y/o procesos.

# CONCEPTO DE SISTEMA DE TIEMPO REAL.

---

- Un sistema de tiempo real es una combinación de uno o varios computadores, dispositivos de hardware de I/O y software de propósito especial, en la que:
  - Hay una fuerte interacción con el entorno
  - El entorno cambia con el tiempo
  - El sistema controla o reacciona de forma simultánea a diferentes aspectos del entorno
- En las aplicaciones de tiempo real, el funcionamiento correcto no sólo depende de los resultados del cálculo, sino también del instante en el que se generan esos resultados
- Como resultado:
  - Se imponen requisitos temporales sobre el software.
  - El software es de naturaleza concurrente.



# CONCEPTO DE SISTEMA DE TIEMPO REAL.

---

- Características esenciales de un sistema de tiempo real:
  - El orden de ejecución de las operaciones depende de:
    - Los eventos externos que recibe del entorno.
    - Del tiempo que transcurre.
  - El cumplimiento de plazos temporales en las respuestas es parte de la especificación funcional.
- De acuerdo con la severidad de los requerimientos temporales:
  - Sistemas de tiempo real estricto (Hard real-time): El incumplimiento de un plazo es un fallo irrecuperable.
  - Sistemas de tiempo real laxos: (Soft real-time): Los requerimientos temporales se cumplen en promedio.

# SISTEMAS EMPOTRADOS.

---

- Un computador (y su software) se considera empotrado si:
  - es un componente integral de un sistema mayor,
  - se usa para controlar, monitorizar o procesar la información de ese sistema
  - y usa dispositivos hardware especiales
- La mayoría de los sistemas de tiempo real son empotrados



## EJEMPLO DE PROGRAMA DE TIEMPO REAL.

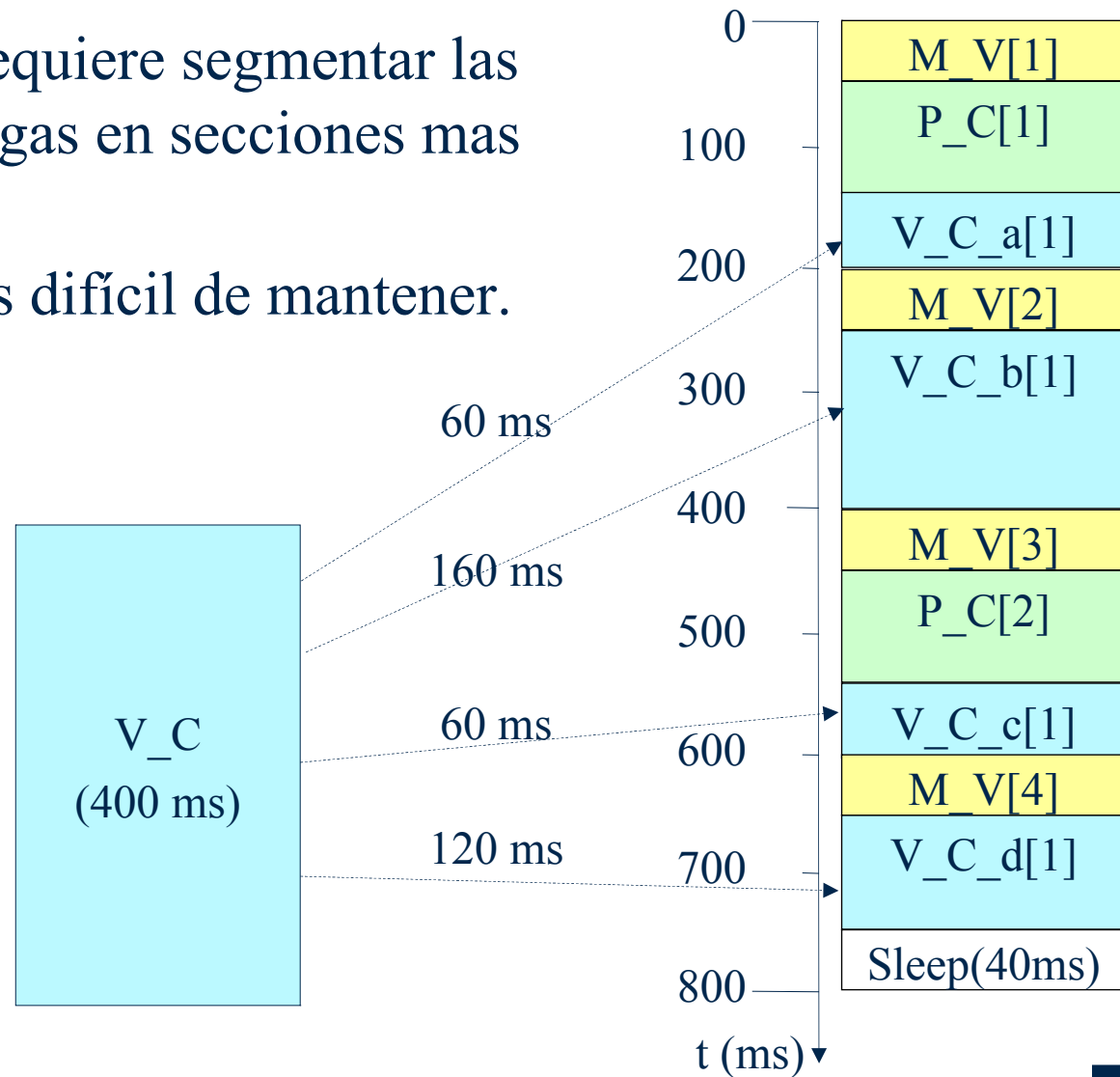
- Software embarcado de control de un coche.

Tarea	Periodo	Duración	% uso
Medida de velocidad	200 ms	40 ms	20%
Control presión carburante	400 ms	100 ms	25%
Control Válvula Carburador	800 ms	400 ms	50%

- El conjunto de tareas hacen uso del 95% de la capacidad del procesador.
- No es posible combinar secuencialmente la tres tareas.

## EJEMPLO: SOLUCIÓN SECUENCIAL.

- La solución requiere segmentar las tareas mas largas en secciones mas breves.
- La solución es difícil de mantener.



## EJEMPLO: TAREA APERIÓDICA.

---

- La tarea C\_S tiene la función de bloquear el cinturón de seguridad si el sensor de aceleración supera un umbral:

Naturaleza:           Aperiódica

Plazo de respuesta:    30 ms

Uso de CPU:            20 ms

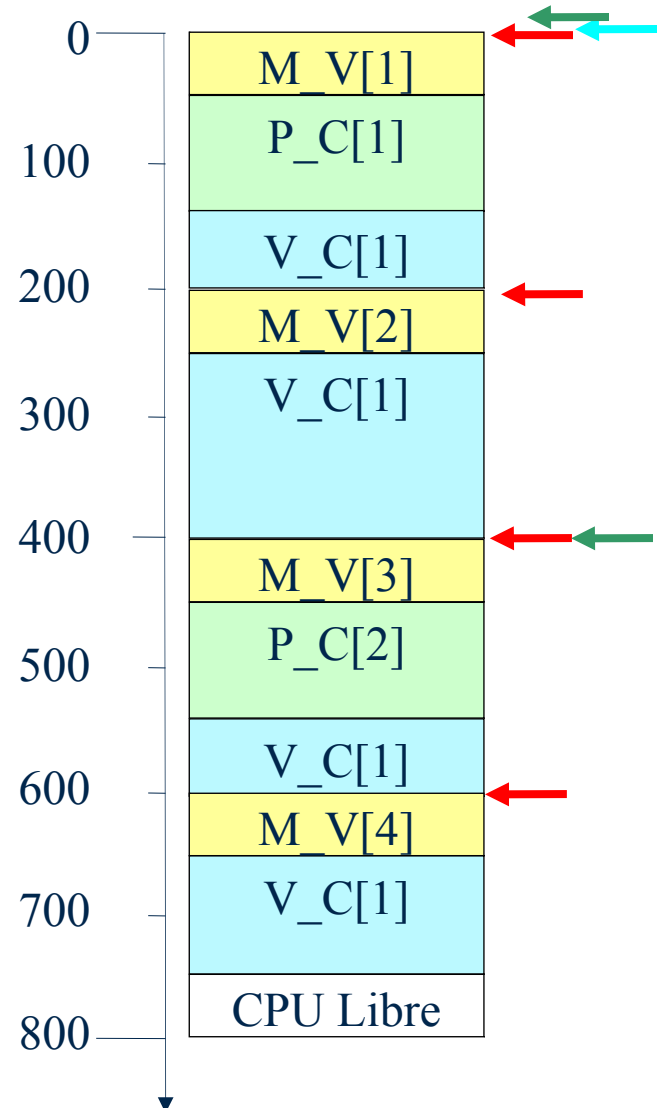
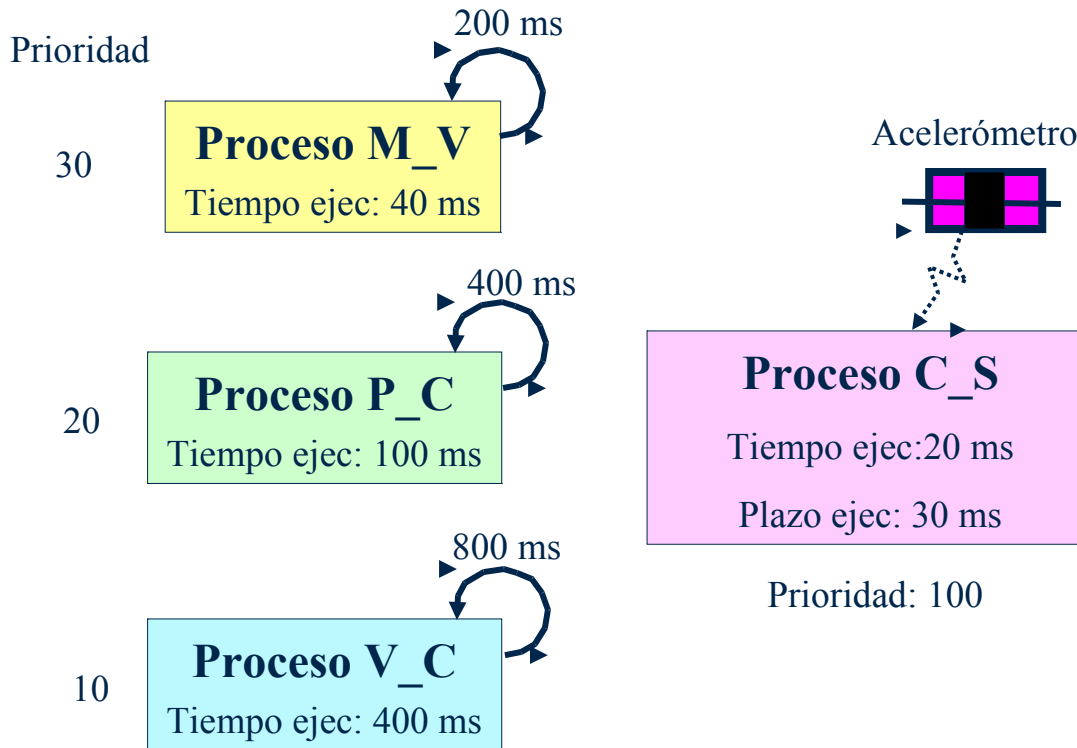
Intervalo mínimo:     800 ms

- Incorporar esta tarea a la estructura secuencial es muy difícil.
- La solución atender el evento en una rutina de interrupción.



# EJEMPLO SOLUCIÓN CONCURRENTENTE.

El programa se plantea como cuatro procesos concurrentes, que solo interaccionan entre sí por compartir el mismo recurso de procesamiento.



# Ejemplo de programa concurrente

```
program Control_coche;
  process P_MedidaVelocidad;    (* Proceso de medida de la velocidad *)
  begin
    repeat M_V; sleep(160 ms); forever;
  end;

  process P_PresiónCarburante;  (* Proceso de control de la presión de carburante *)
  begin
    repeat P_C; sleep(300 ms); forever;
  end;

  process P_ControlValvulaCarburador; (* Proc. de control de válvula del carburador *)
  begin
    repeat V_C; sleep(400 ms); forever;
  end;

  process P_ControlCinturan;    (* Proceso de atención del control del cinturón *)
  begin
    repeat wait evento; C_S; forever;
  end;

begin          (* Programa principal *)
  cobegin
    P_MedidaVelocidad;          P_PresionCarburante;
    P_ControlValvulaCarburador; P_ControlCinturón;
  coend;
end.
```