

DAM - Programación de servicios y procesos

Tema 2 - Programación multiproceso

Roberto Sanz Requena

rsanz@florida-uni.es

Índice

1. **Introducción**
2. **Ejecutables, procesos y servicios**
3. **Estados de un proceso**
4. **Hilos**
5. **Programación concurrente**
6. **Programación paralela y distribuida**
7. **Creación de procesos**
8. **Comunicación entre procesos**
9. **Gestión de procesos**
10. **Sincronización entre procesos**

1. Introducción

Multiprocesamiento o multiproceso

- Uso de dos o más procesadores (CPU) en una computadora para la ejecución de uno o varios procesos (programas corriendo).
- Comparten en parte memoria principal y periféricos.

Multitarea

- Ejecución concurrente de segmentos de múltiples tareas (de uno o varios procesos) de forma entrelazada (cambios de contexto) en una única CPU.
- No se ejecutan tareas en paralelo en el mismo tiempo exactamente, sino que se permite que varias tareas avancen en un periodo de tiempo dado.

Importante: definición del **problema** y disponibilidad de **recursos**

2. Ejecutables, procesos y servicios

Ejecutable

- Ejecutable, o archivo ejecutable, es un archivo binario cuyo contenido se interpreta por el ordenador como un programa.
- Se ejecutan y controlan por el sistema operativo.
- Ejemplos: Windows son los .EXE (Windows), .COM (MS-DOS), .JAR o .CLASS (Java).
- Determinar si un archivo es ejecutable es sobre todo una cuestión de convención. Unos sistemas operativos se basan en la extensión de archivo (como la terminación .exe) y otros lo hacen leyendo los metadatos (como los bits de permiso de ejecución en Unix).
- En general, los archivos ejecutables son el principal medio de transmisión de virus y malware.

2. Ejecutables, procesos y servicios

Ejecutable en Java

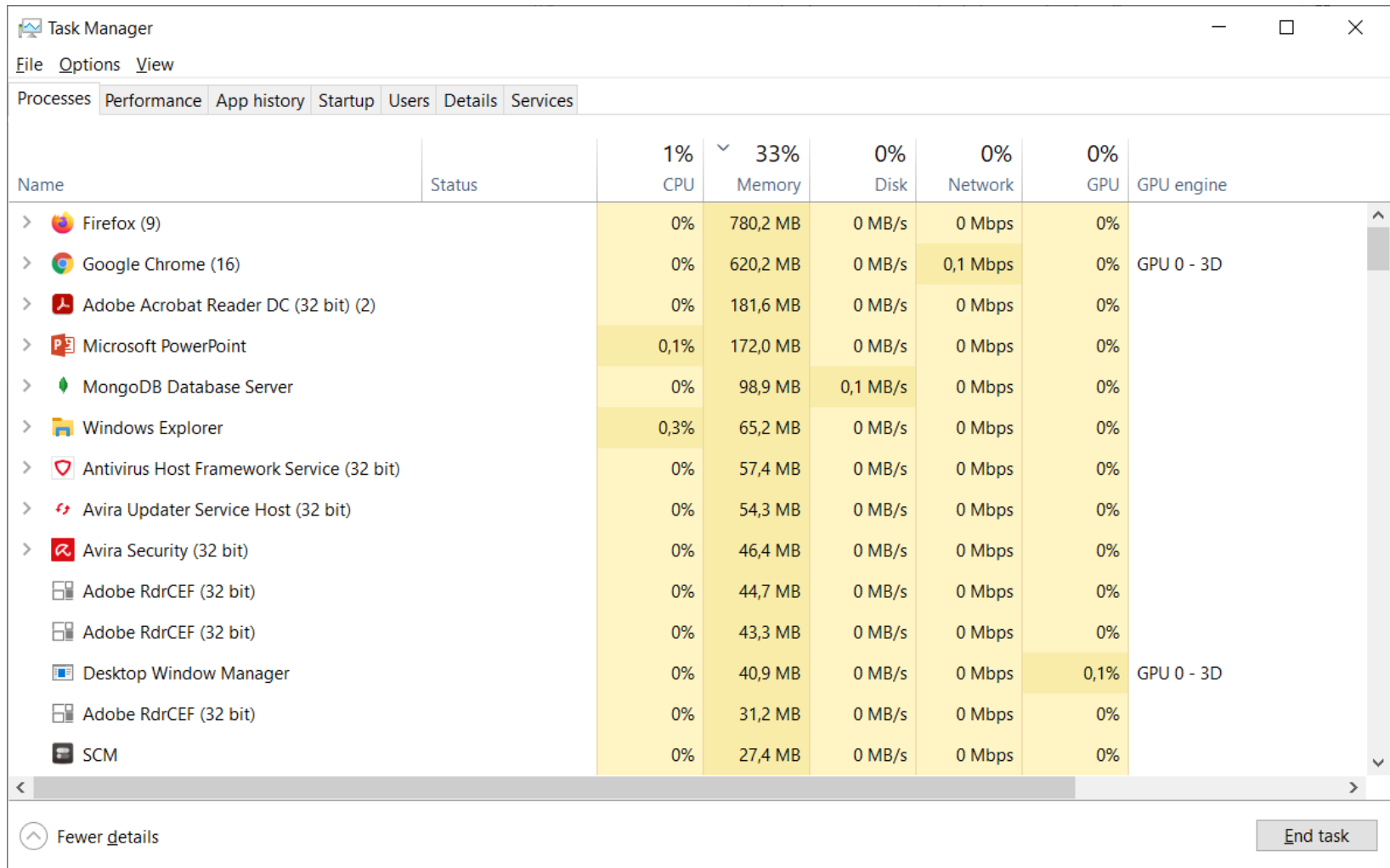
- Los archivos Java compilados (.class o .jar) contienen “instrucciones virtuales de Java”, no ejecutables por un microprocesador.
- Un intérprete (máquina virtual) toma el programa, lo interpreta y lo traduce a instrucciones del microprocesador concreto.
- Ese proceso se hace “al instante” o JIT (Just-In-Time).
- Comparando con un .exe, Java puede llegar a ser más eficiente si tiene en cuenta las instrucciones más actualizadas que admiten los microprocesadores.

2. Ejecutables, procesos y servicios

Procesos

- Instancia de una aplicación (ejecutable) que está en ejecución en el sistema operativo.
- Una aplicación puede tener varios procesos en ejecución simultánea.
- Estados posibles:
 - Ejecución (consumo de microprocesador).
 - Pausa/espera (debe seguir en ejecución pero espera que el microprocesador lo retome).
 - Interrumpido (el usuario u otro evento interrumpe su ejecución sin que haya terminado).
- Reserva de recursos: memoria RAM (registros y variables).

2. Ejecutables, procesos y servicios



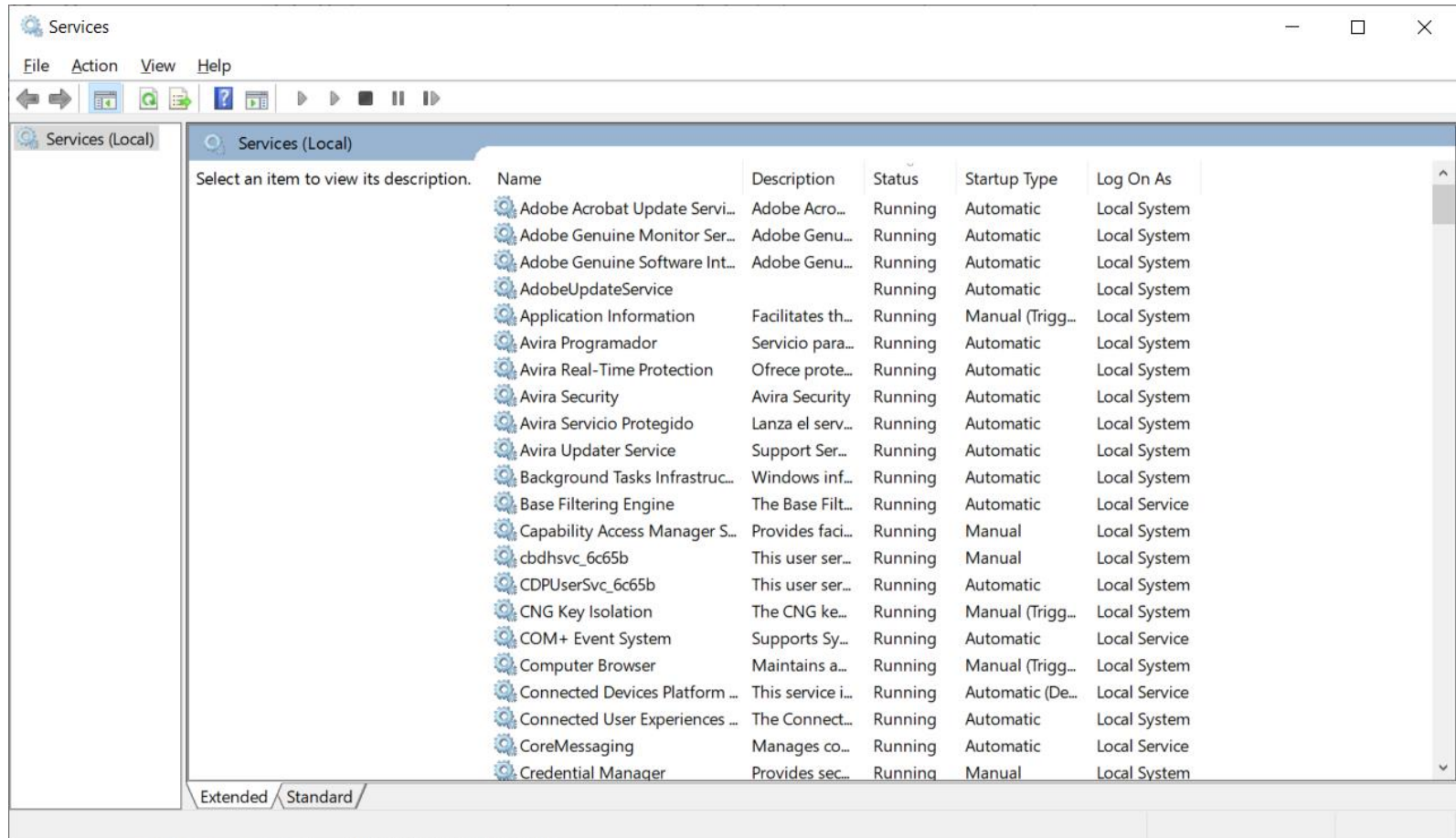
Name	Status	1% CPU	33% Memory	0% Disk	0% Network	0% GPU	GPU engine
> Firefox (9)		0%	780,2 MB	0 MB/s	0 Mbps	0%	GPU 0 - 3D
> Google Chrome (16)		0%	620,2 MB	0 MB/s	0,1 Mbps	0%	
> Adobe Acrobat Reader DC (32 bit) (2)		0%	181,6 MB	0 MB/s	0 Mbps	0%	
> Microsoft PowerPoint		0,1%	172,0 MB	0 MB/s	0 Mbps	0%	
> MongoDB Database Server		0%	98,9 MB	0,1 MB/s	0 Mbps	0%	
> Windows Explorer		0,3%	65,2 MB	0 MB/s	0 Mbps	0%	GPU 0 - 3D
> Antivirus Host Framework Service (32 bit)		0%	57,4 MB	0 MB/s	0 Mbps	0%	
> Avira Updater Service Host (32 bit)		0%	54,3 MB	0 MB/s	0 Mbps	0%	
> Avira Security (32 bit)		0%	46,4 MB	0 MB/s	0 Mbps	0%	
Adobe RdrCEF (32 bit)		0%	44,7 MB	0 MB/s	0 Mbps	0%	
Adobe RdrCEF (32 bit)		0%	43,3 MB	0 MB/s	0 Mbps	0%	GPU 0 - 3D
Desktop Window Manager		0%	40,9 MB	0 MB/s	0 Mbps	0,1%	
Adobe RdrCEF (32 bit)		0%	31,2 MB	0 MB/s	0 Mbps	0%	
SCM		0%	27,4 MB	0 MB/s	0 Mbps	0%	

2. Ejecutables, procesos y servicios

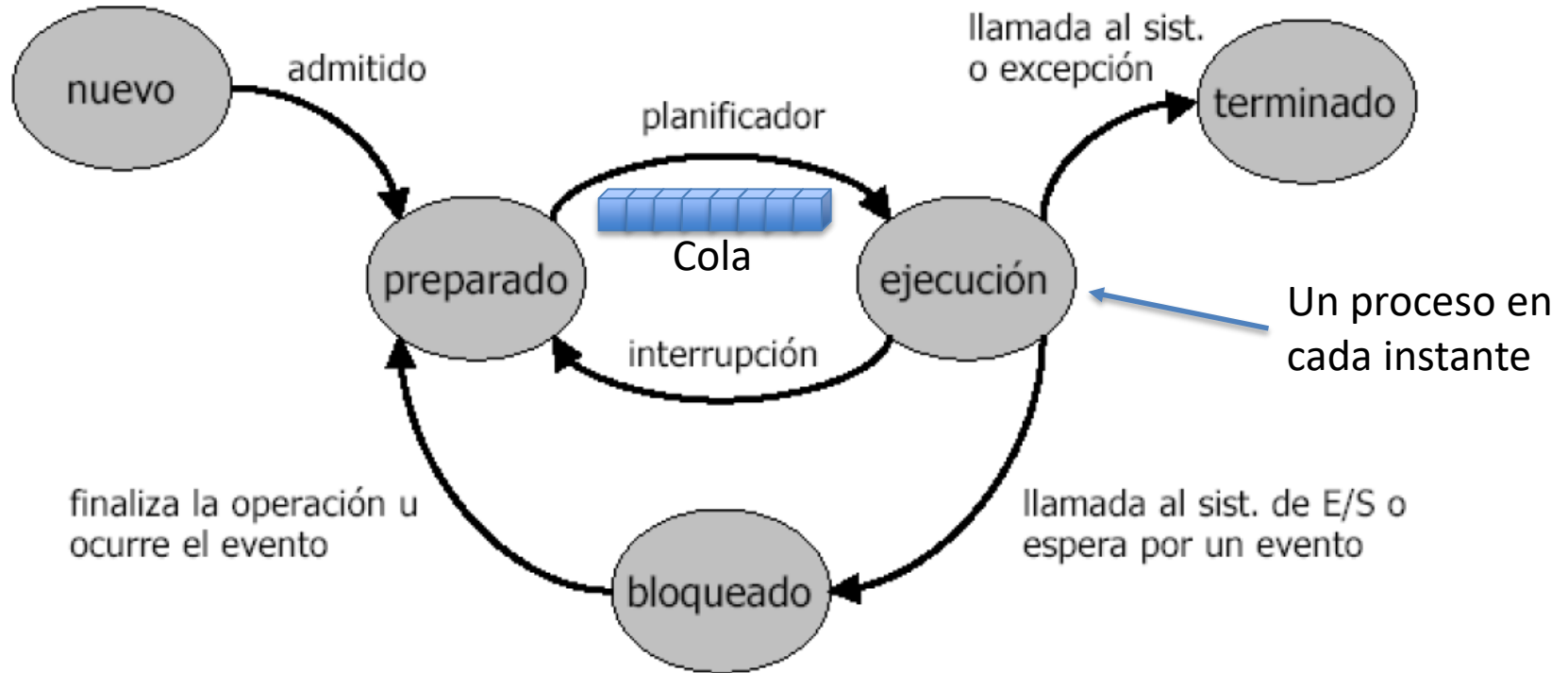
Servicios

- Proceso que se ejecuta en segundo plano y no requiere interacción con el usuario.
- Es un programa que proporciona funcionalidades a otro programa (por ejemplo el servicio de impresión, la conexión a red, etc.).
- Servicios web vs. Servicios locales: acceso a datos, procesos, etc. de un ordenador remoto.

2. Ejecutables, procesos y servicios



3. Estados de un proceso

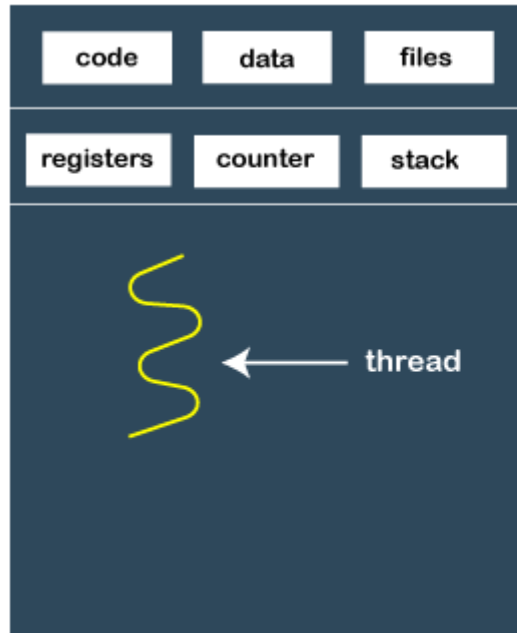


4. Hilos

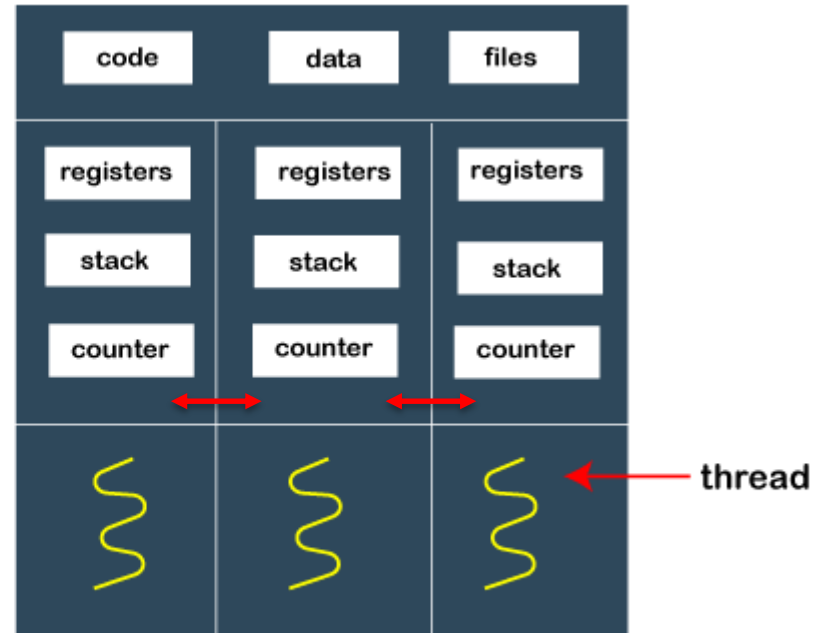
- Un **hilo** es un concepto más avanzado que un proceso.
- En procesos...
 - Cada proceso tiene su propio espacio en memoria.
 - Si abrimos 20 procesos cada uno de ellos consume 20x de RAM
 - Un proceso no tiene acceso a los datos de otro procesos.
- Un hilo es un proceso mucho más ligero...
 - El código y los datos se comparten de una forma distinta.
 - Un hilo puede acceder a los datos de otro hilo.
 - Más eficiencia en recursos y computación
 - Pueden ser más complejos de programar.

IMPORTANTE:
Definición del problema

4. Hilos



Single-threaded process

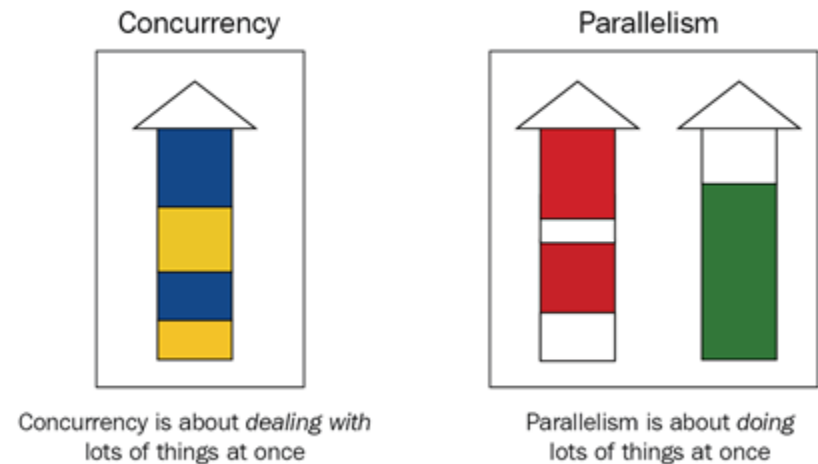
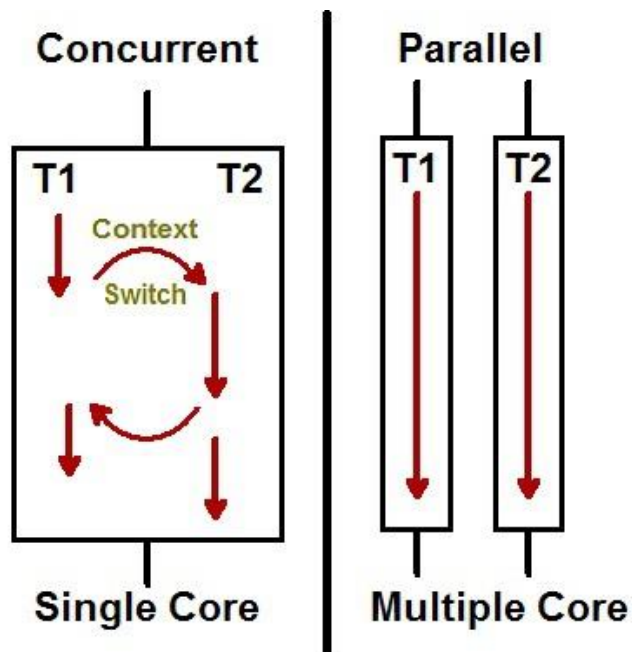


Multi-threaded process

<https://www.javatpoint.com/process-vs-thread>

5. Programación concurrente

- Programas que pueden tener varios procesos/hilos que colaboran para ejecutar un trabajo.

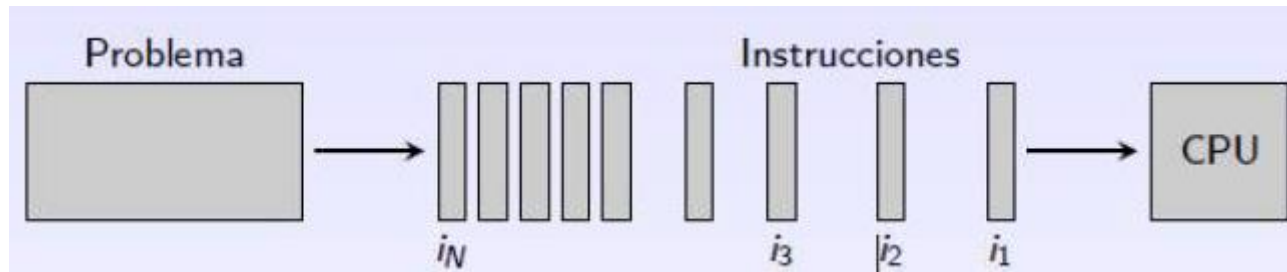


Concurrencia = diseño
Paralelismo = hardware

<https://www.codeproject.com/Articles/1267757/Concurrency-vs-Parallelism>
<https://medium.com/@PacktExprtNtwrk/concurrency-and-parallelism-8526beb149f2>

6. Programación paralela y distribuida

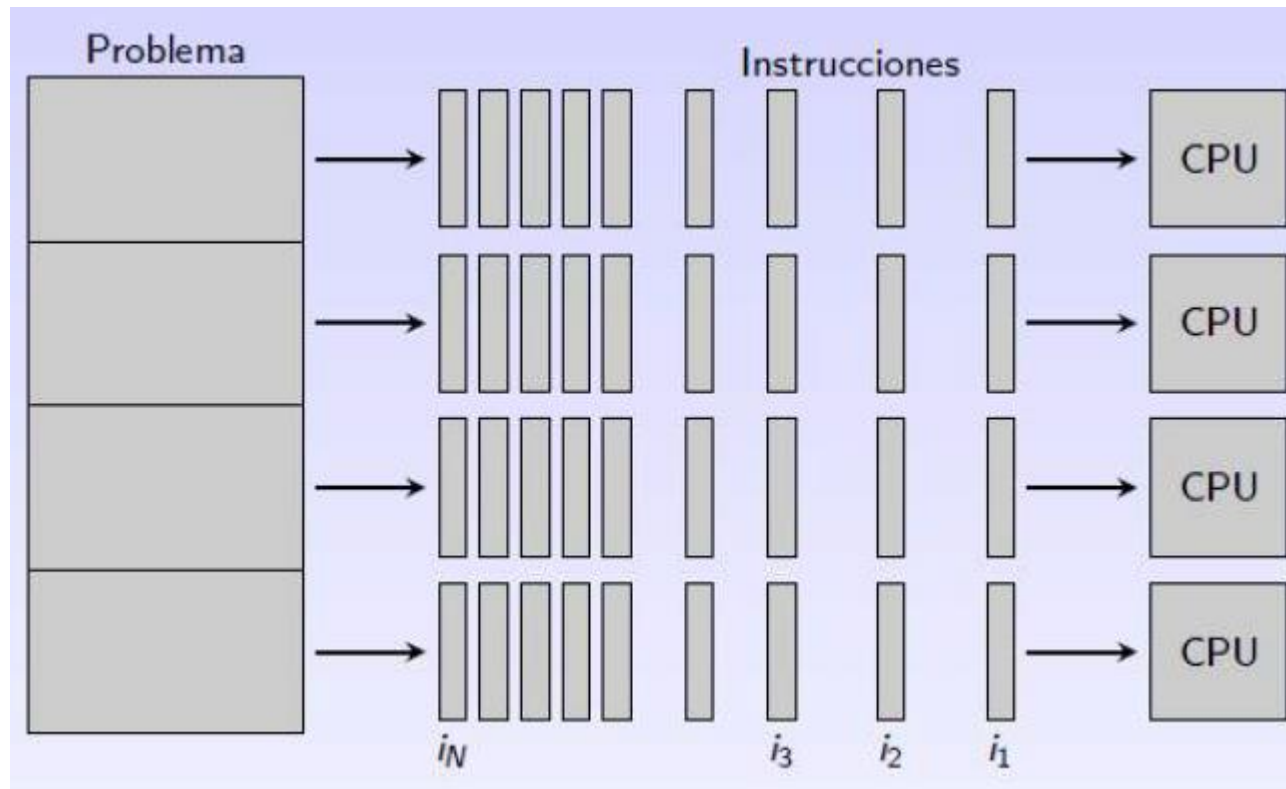
- Programación tradicional: orientación a cálculo secuencial (serie)



- Programación paralela: orientación a cálculo en paralelo
 - Varias CPUs / núcleos.
 - División del problema en partes independientes.
 - Ejecución simultánea de las partes.

6. Programación paralela y distribuida

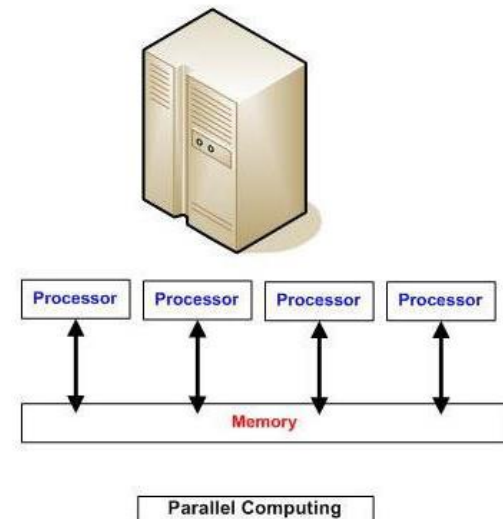
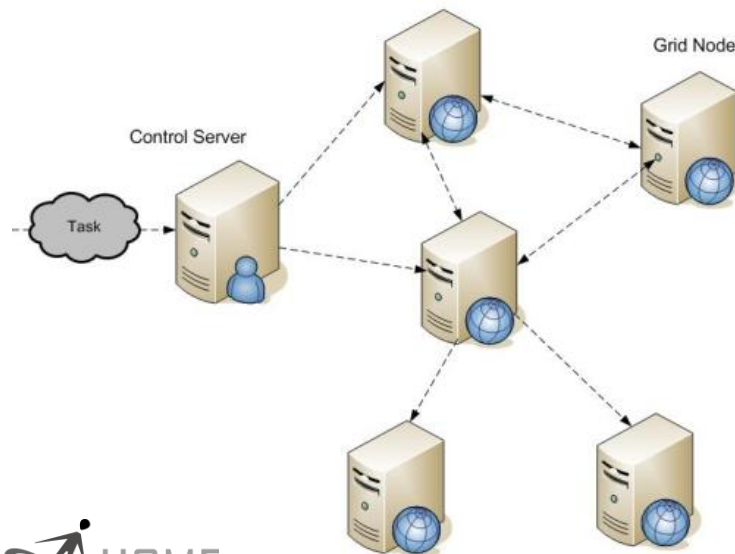
- Programación paralela:



Parallel
Virtual
Machine

6. Programación paralela y distribuida

- Programación distribuida: ejecución en varios ordenadores en red



<https://apuntescomputacion.wordpress.com/2008/08/16/%C2%BFque-es-y-como-funciona-un-grid/>
https://www.researchgate.net/publication/274284241_The_Economics_of_Cloud_Computing_on_Educational_Services

6. Programación paralela y distribuida

- Grid computing (programación distribuida) vs cloud computing
- Grid computing:
 - Arquitectura distribuida (equipos heterogéneos).
 - El usuario proporciona recursos y consume recursos.
 - Recursos gestionados de forma distribuida y colaborativa.
 - Escalabilidad media.
- Cloud computing:
 - Arquitectura cliente-servidor.
 - El usuario (cliente) sólo necesita conexión a Internet.
 - Contratación de servicios y recursos.
 - Recursos gestionados de forma centralizada (p.ej. Amazon, Azure, etc.).
 - Alta escalabilidad.

6. Programación paralela y distribuida

¿Más es mejor? Depende...

Frecuencia de reloj (GHz), número de núcleos (4, 8, 12,...), hyperthreading,...

Para aprovechar las posibilidades de los sistemas multinúcleo y HT hay que tener en cuenta que las aplicaciones deben estar optimizadas para ese hardware.

En la actualidad las aplicaciones que necesitan utilizar el procesamiento de manera intensiva se diseñan para aprovechar al máximo las prestaciones de la CPU y la GPU.

La elección de un sistema u otro depende del uso que se le vaya a dar.

<https://www.guru99.com/cpu-core-multicore-thread.html>

<https://www.cpubenchmark.net/singleCompare.php>

<https://www.pcworld.com/article/3039552/tested-how-many-cpu-cores-you-really-need-for-directx-12-gaming.html>

7. Creación de procesos

Creación de un proceso en Java: lanzar la aplicación Calculadora

```
public class LanzadorProcesos {
    public void ejecutar(String ruta){
        ProcessBuilder pb;
        try {
            pb = new ProcessBuilder(ruta);
            pb.start();
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }

    public static void main(String[] args) {
        String ruta = "C:\\\\Windows\\\\System32\\\\calc.exe";
        LanzadorProcesos lp = new LanzadorProcesos();
        lp.ejecutar(ruta);
        System.out.println("Finalizado");
    }
}
```

7. Creación de procesos

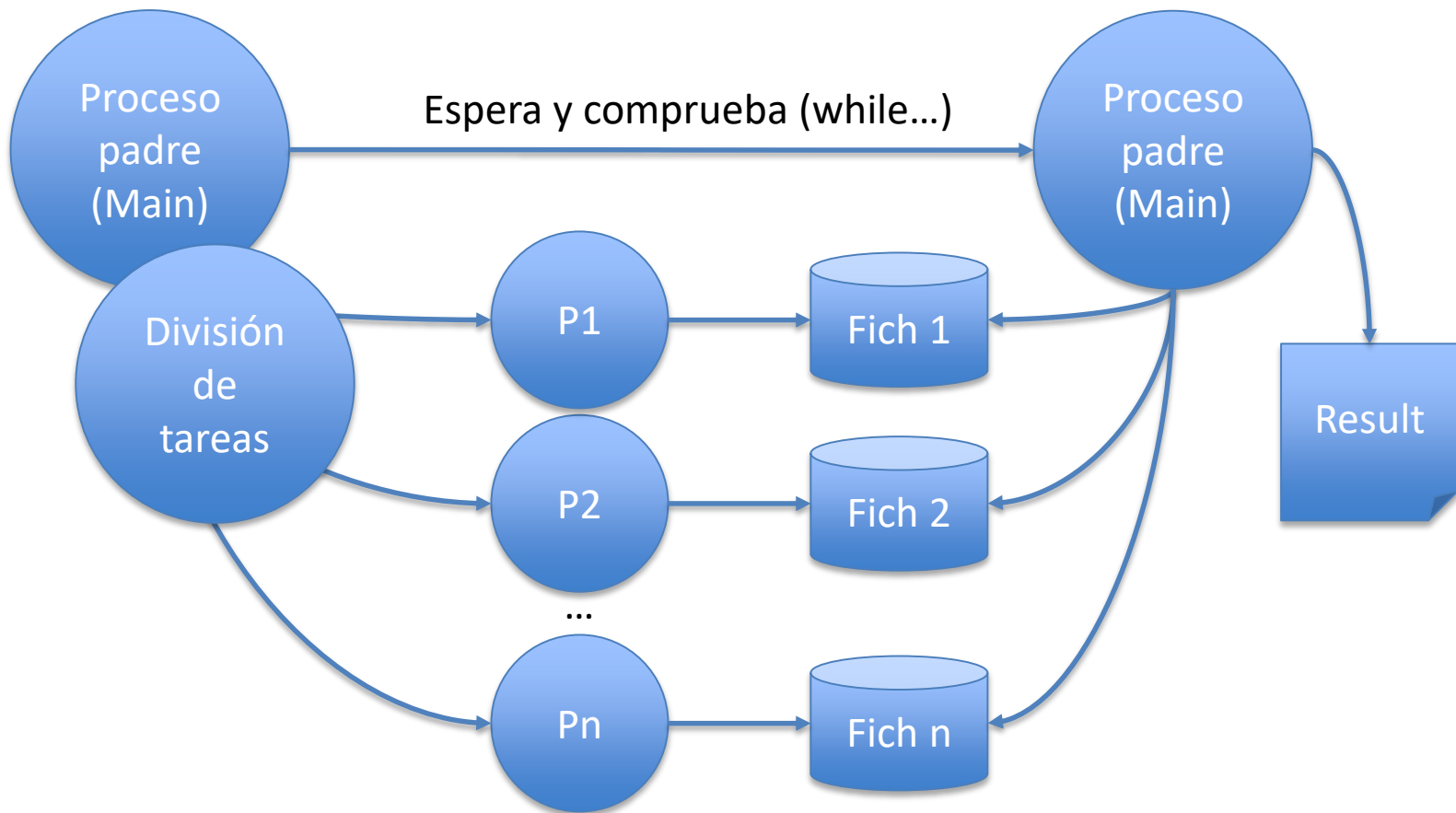
Creación de un proceso en Java: sumador

1. Crear una clase Java para sumar todos los números comprendidos entre dos valores, incluyendo ambos valores.
2. Crear clase `Sumador` y otra clase `Lanzador` para lanzar varios procesos que instancien cada uno a la clase `Sumador`. Suma de 1 a 100 en 2 procesos (1 a 50 y 51 a 100).

Ejemplo “PSP_MultiprocesoSumador”

8. Comunicación entre procesos

- Utilizar ficheros para escribir el resultado parcial de cada proceso hijo.
- Utilizar el proceso padre para recuperar todos los resultados.



8. Comunicación entre procesos

```
File directorioSumador = new File("...");
File fichResultado = new File(fichResultados);
String clase = "es.florida.multiproceso.Sumador";
String javaHome = System.getProperty("java.home");
String javaBin = javaHome + File.separator + "bin" + File.separator + "java";
String classpath = System.getProperty("java.class.path");
String className = clase;

List<String> command = new ArrayList<>();
command.add(javaBin);
command.add("-cp");
command.add(classpath);
command.add(className);
command.add(String.valueOf(n1));
command.add(String.valueOf(n2));

ProcessBuilder builder = new ProcessBuilder(command);
builder.directory(directorioSumador);
builder.redirectOutput(fichResultado);
Process p = builder.start();
```

Ejemplo "PSP_MultiprocesoSumador_v2"

9. Gestión de procesos

- **Windows:** Administrador de tareas (Task Manager)

The left screenshot shows the Windows Task Manager 'Processes' tab. A context menu is open for 'Adobe Acrobat Reader DC (32 bit)', with the 'Go to details' option highlighted. The right screenshot shows the 'Details' tab, displaying a list of running processes with columns for Name, PID, Status, User name, CPU, CPU time, and Memory (active ...).

Name	PID	Status	User name	CPU	CPU time	Memory (active ...)
AcroRd32.exe	7176	Running	R	00	0:00:02	31.216 K
AcroRd32.exe	4428	Running	R	00	0:00:57	153.896 K
AdobeNotificationCli...	10376	Suspended	R	00	0:00:00	0 K
AdobeUpdateService...	3600	Running	SYSTEM	00	0:00:00	656 K
AGMService.exe	3608	Running	SYSTEM	00	0:00:01	2.252 K
AGSService.exe	3568	Running	SYSTEM	00	0:00:00	1.608 K
ApplicationFrameHo...	1508	Running	R	00	0:00:02	5.188 K
armsvc.exe	3500	Running	SYSTEM	00	0:00:00	548 K
avguard.exe	3508	Running	SYSTEM	00	0:04:01	16.064 K
Avira.SoftwareUpdat...	3532	Running	SYSTEM	00	0:00:15	30.268 K
Avira.Spotlight.Servic...	3524	Running	SYSTEM	00	0:00:27	38.164 K
Avira.Spotlight.Systra...	6904	Running	R	00	0:00:01	732 K
avshadow.exe	1156	Running	SYSTEM	00	0:00:00	872 K
Calculator.exe	10352	Suspended	R	00	0:00:01	0 K
ClassicStartMenu.exe	7284	Running	R	00	0:00:00	1.300 K
conhost.exe	2932	Running	SYSTEM	00	0:00:00	5.460 K
csrss.exe	664	Running	SYSTEM	00	0:00:02	952 K

9. Gestión de procesos

- **Linux:** línea de comandos (console)
 - Lanzar proceso: instrucción correspondiente al proceso (permisos)
 - Arrancar servicio: `sudo service <servicio> <start/stop/restart>`
 - Ver lista e información de procesos: `ps / ps aux / top`
 - Detener proceso por nºproceso: `kill -9 <nºproceso>`
 - Detener proceso por nombre: `killall -9 <nombre_proceso>`
 - Enviar último proceso a segundo plano (background): `bg`
 - Traer último proceso a primer plano (foreground): `fg`
 - Asignar prioridad: `nice -n <prioridad> <nombre_proceso>`
 - 20 → más prioridad
 - ...
 - 0 → prioridad por defecto
 - ...
 - 19 → menos prioridad

10. Sincronización entre procesos

- Mecanismo para coordinar las actividades de dos o más procesos.
- Acceso a recursos compartidos cuya integridad sea crítica (p.ej. escritura en fichero, hilo en espera de resultados de otro hilo).
- Java: **monitor** para cada objeto instanciado.
 - Permite bloquear el acceso al objeto o a alguno de sus métodos.
 - Uso de la palabra `synchronized` como modificador del método.
 - Cuando un hilo accede al método, el monitor lo bloquea y ningún otro hilo puede acceder hasta que no finaliza el primer hilo.

Actividad Entregable 2 - Multiproceso

Presentación de la Actividad Entregable 2 (AE2_T2_Multiproceso)