TRABAJO PRÁCTICO N°1 METODOLOGÍA DE LA INVESTIGACIÓN

Alumno: Magiarate Funes Mayra

PROGRAMACIÓN CONCURRENTE

INTRODUCCIÓN

Cuando Tratamos de entender lo que es la Programación concurrente no podemos dejar de hablar de concurrencia; concurrencia es cuando ocurren varios sucesos de manera contemporánea (hablamos de contemporaneidad y no de simultaneidad).

En base a esto, la concurrencia en computación está asociada a la "ejecución" de varios procesos que coexisten temporalmente.

Para definirla correctamente, debemos diferenciar entre programa y proceso.

Programa: Conjunto de sentencias/instrucciones que se ejecutan secuencialmente. Se asemeja al concepto de clase dentro de la POO. Es por tanto un concepto estático.
Programa concurrente: Ejecución de acciones simultáneamente.
Programa paralelo: Programa que se ejecuta en un sistema multiprocesador.
Programa distribuido: Programa paralelo para ejecutarse en sistemas distribuidos.
Proceso: Básicamente, se puede definir como un programa en ejecución. Líneas de código en ejecución de manera dinámica. Se asemeja al concepto de objeto en POO.

La concurrencia aparece cuando dos o más procesos son contemporáneos. Un caso particular es el paralelismo (programación paralela).
Este podría ser un ejemplo de paralelismo o concurrencia entre los procesos Po y P1, quedando bien claro que son procesos contemporáneos pero no simultáneos puesto que hay espera. Debemos tener en cuenta que el ejemplo es para el caso de contar con un solo procesador.

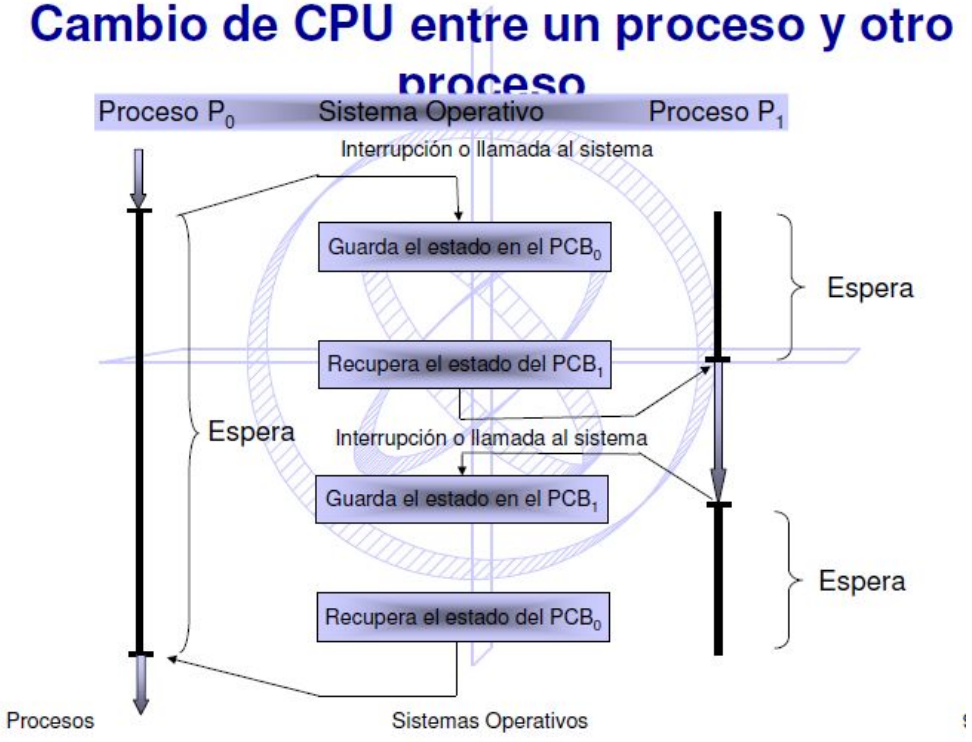


Figura nº1

Los procesos pueden "competir" o colaborar entre sí por los recursos del sistema. Por tanto, existen tareas de colaboración y sincronización.

¿Pero entonces qué es la Programación Concurrente?

Cuando hablamos de Programación concurrente hablamos de las técnicas que ha utilizado el programador Ingeniero para resolver los problemas de comunicación y sincronización entre procesos que llevará a cabo el sistema operativo.

La programación concurrente se encarga del estudio de las nociones de ejecución concurrente, así como sus problemas de comunicación y sincronización.

Dicho de otra manera: Se conoce por programación concurrente a la rama de la informática que trata de las **técnicas de programación** que se usan para expresar el paralelismo entre tareas y **para resolver los problemas de comunicación y sincronización entre procesos**.

El principal problema de la programación concurrente corresponde a no saber en qué orden se ejecutan los programas (en especial los programas que se comunican). Se debe tener especial cuidado en que este orden no afecte el resultado de los programas.

¿Cuáles Son las Técnicas o Cómo hace la programación concurrente?

El Ingeniero puede subdividir un programa en procesos, éstos se pueden "repartir" entre procesadores o gestionar en un único procesador según importancia.

O bien seguir o aplicar las siguientes Metodologías:

- Sistemas de control: Captura de datos, análisis y actuación (p.ej. sistemas de tiempo real).
- Tecnologías web: Servidores web que son capaces de atender varias peticiones concurrentemente, servidores de chat, email, etc.
- Aplicaciones basabas en GUI: El usuario hace varias peticiones a la aplicación gráfica (p.ej. Navegador web).
- Simulación: Programas que modelan sistemas físicos con autonomía.
- Sistemas Gestores de Bases de Datos: Cada usuario un proceso.

¿Cómo se aplica entonces la programación concurrente a distintos tipos de Hardware?

En un **Sistemas monoprocesador**, podemos tener concurrencia, gestionando el tiempo de procesador para cada proceso (Como se muestra en la figura nº1).

En **Sistemas multiprocesador.** Un proceso en cada procesador (Como en la figura nº2). Éstos pueden ser de memoria compartida (fuertemente acoplados) o con memoria local a cada procesador (débilmente acoplados). Un ejemplo muy conocido y útil son los sistemas distribuidos (por ej. Beowulfs).

En relación a la concurrencia se pueden clasificar en aquellos que funcionan con variables/memoria compartida o paso de mensajes.

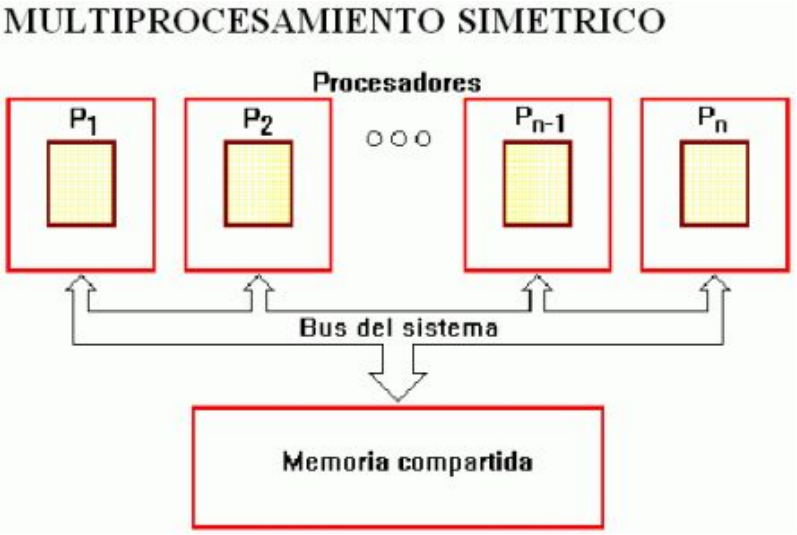


Figura nº2.

Características de los sistemas concurrentes

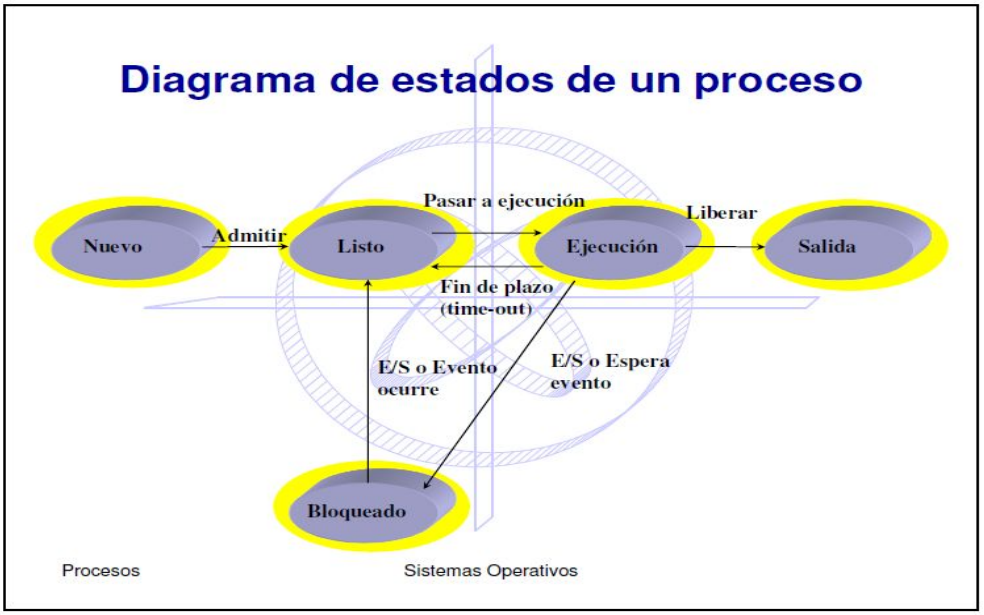
- ✓ Orden de ejecución: A diferencia de los programas secuenciales el flujo del programa sigue un orden parcial. Ante una misma entrada no se sabe cuál va a ser el orden seguido.
- ✓ Indeterminismo: El orden parcial produce consecuentemente un comportamiento indeterminista. Es decir, repetidas ejecuciones sobre un mismo conjunto de datos resultan "diferentes resultados".

Problemas inherentes a los sistemas concurrentes

- ✓ Exclusión mutua: Como lo que realmente se ejecuta concurrentemente son las instrucciones de ensamblador, cuando se comparten variables se excluyen los valores. Por ejemplo, dos bucles que hacen x=x+1.
- ✓ Condición de sincronización: La necesidad de coordinar los procesos.

Pero entonces ¿Cómo se hace qué se debe tener en cuenta?

Ciclo de vida de un Proceso



Entonces debemos tener en cuenta:

- ✓ Disposición de memoria de un proceso (Básicamente, existe un espacio de usuario y un espacio de núcleo).
- ✓ Cada proceso tiene sus propias características: código, variables, id, contadores, pila, etc. Son monohilo.
- ✓ La gestión y el cambio de contexto de cada proceso es muy costoso. Se debe actualizar los registros de uso de memoria, y controlar los estados en los que quedan los procesos.

¿Qué se puede ejecutar concurrentemente?

x=x+1;
y=x+1;
La primera instrucción se debe ejecutar antes de la segunda!!
x=1;y=2;z=3; El orden no interviene en el resultado final!!

Condiciones de Bernstein

Para que dos conjuntos de instrucciones se puedan ejecutar concurrentemente se tiene que cumplir que:

- La intersección entre el conjunto de variables leidas por uno, con el conjunto de variables que escribe el otro, debe ser nulo. Y viceversa.
- La intersección del conjunto de variables que escribe uno y el conjunto que escribe el otro, debe ser nulo.

Ejemplo: Denotamos por L(lectura) y E(escritura)

P1: a=x+y;	L(P1)={x,y}	E(P1)={a}	Se debe cumplir:
P2: b=z-1;	L(P2)={z}	E(P2)={b}	
P3: c=a-b;	L(P3)={a,b}	E(P3)={c}	$L(Pi) \Omega E(Pj)=\varnothing;$
P4: w=c+1;	L(P4)={c}	E(P4)={w}	$E(Pi) \Omega L(Pj)=\varnothing;$
			$E(Pi) \Omega E(Pj)=\varnothing;$

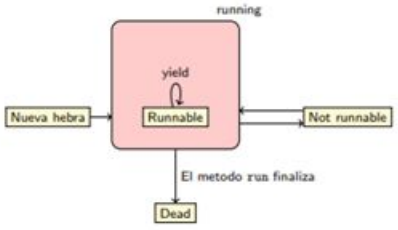
Para finalizar podemos dejar un buen ejemplo de Programación concurrente, como es la programación en java usando threads, y estaríamos hablando así de programas concurrentes que utilizan la programación concurrente.

Threads

Implementado el interfaz `java.lang.Runnable`.

```
public class PrRunnable implements Runnable {
    public final void run() {
        Thread hebra = Thread.currentThread();
        boolean sigue=true;
        for (int i=0; i<100 && sigue; i++) {
            try {
                System.out.println(hebra.getName()+":"+i);
                hebra.sleep(20);
            } catch (InterruptedException e) {
                System.out.println(hebra.getName()+" interrumpida");
                sigue=false;
            }
        }
    }
    public static final void main(final String[] args) {
        Thread p = new Thread(new PrRunnable(),"mia");
        p.start();
    }
}
```

Ciclo de vida de una hebra



Bibliografía:

Miguel Ángel Rodríguez Florido:
https://www2.ulpgc.es/hege/almacen/download/20/20233/tema1.pdf

Programación concurrente:
https://www.fing.edu.uy/tecnoinf/mvd/cursos/so/material/teo/so07-concurrencia.pdf

Procesos: Diapositivas y figuras de la cátedra Arquitectura y sistemas Operativos, Ingeniero Tonelli.