Sesión 2

Concurrencia en el servidor

Sesión 2: Eco Concurrente

Objetivo: Dotar de concurrencia al servidor eco de la sesión anterior.

1.- Usando **`fork()`**

2.- Usando **`select()`**

3.- Usando **`hilos.`**

Creación previa de procesos

Crear 5 hijos antes de entrar en el bucle de espera de clientes.

Problema:

Máquinas en las que la exclusión mutua entre procesos no la realiza el S.O. Todos los procesos creados acceden a la vez a **accept()** y retornan con un error ya que no hay conexiones esperando.

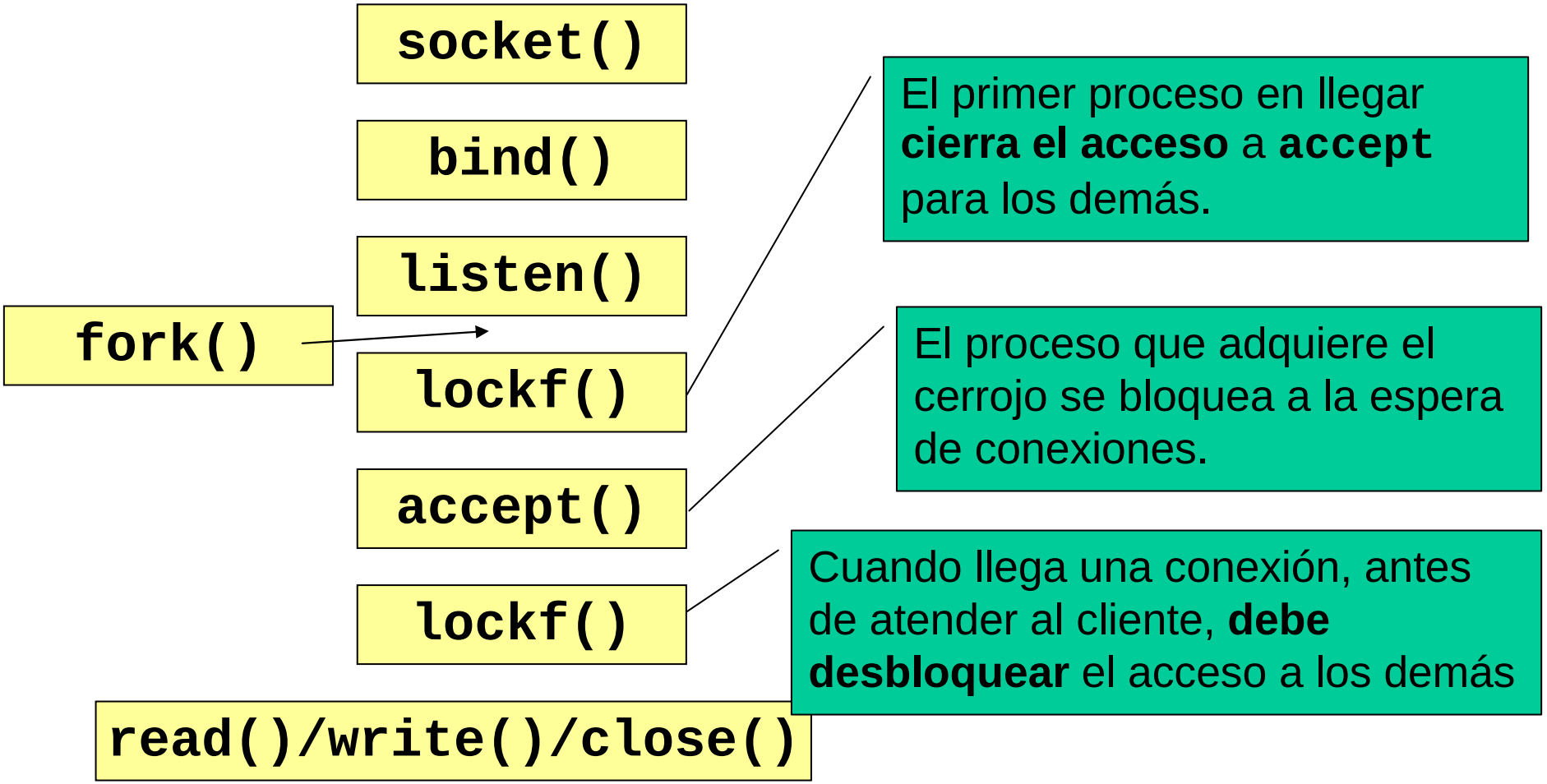
Ejemplo de S.O. con este problema: Solaris.

Solución:

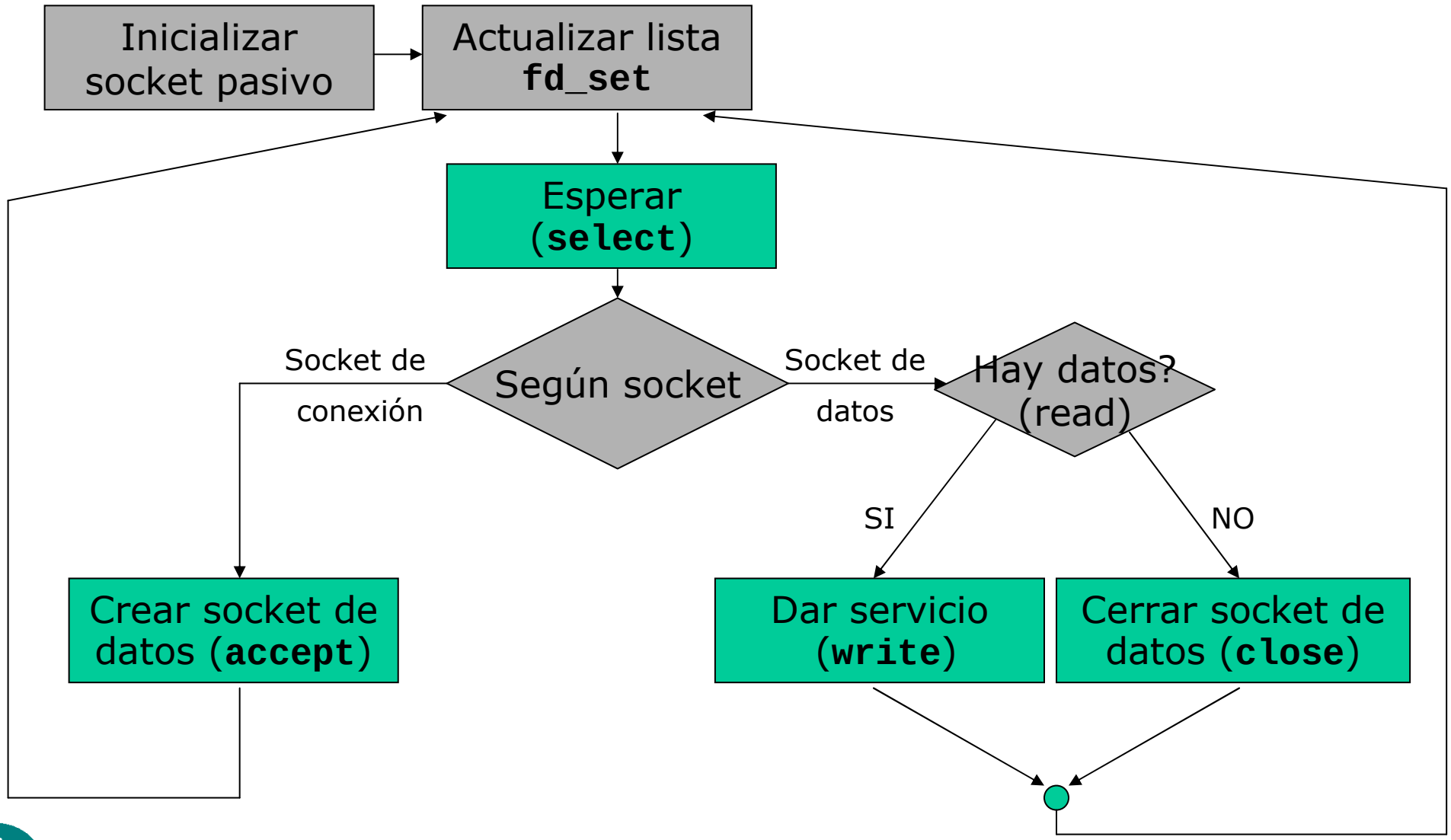
Construir la exclusión mutua “a mano” usando **lockf()**.

Ejercicio 1: Usando fork

Secuencia de llamadas



Ejercicio 2: Usando select()



Ejercicio 3: Usando hilos

Funciones:

```
ret=pthread_create(hilo, attr,
                   rutina_inicio, param);

pthread_t *hilo; pthread_attr_t *attr;

void *(*rutina_inicio)(void*); void *param;

int ret;


pthread_exit(puntero_a_dato);
   void *puntero_a_dato;
```

Ejercicio 3: Usando hilos

Precauciones

- Usando threads, todos los hilos corren en el mismo proceso. Las variables globales son compartidas.
- No debe accederse a las variables globales sin mecanismos de exclusión mutua (funciones pthread_mutex*)
- Es más sencillo usar siempre que se pueda variables locales (de la función lanzada en el hilo)