# *Prácticas de Sistemas operativos*

David Arroyo Guardeño

Escuela Politécnica Superior de la Universidad Autónoma de Madrid

## Cuarta semana: Hilos

# *Segunda práctica*

1. Hilos: ejercicios 3 y 4
2. Señales: ejercicios 6, 8 y 10

Jueves 5 de marzo

# *Entregas*

- Ejercicio3
- Ejercicio 4

# *¿Qué son los hilos?*

- ✓ Como los procesos $\rightarrow$ ejecución concurrente de múltiples tareas
- ✓ Un proceso puede contener múltiples hilos
  - ✗ Todos se ejecutan independientemente
  - ✗ Comparten la misma memoria global
    1. Datos inicializados
    2. Datos sin inicializar
    3. Espacio de direcciones

```c
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define NUM_HILOS 5

int l = 0;

void * funcion_hilo(void *id)
{
  int i;

  for(i=0;i<50;i++){
    printf("Hilo %d: i=%d, l=%d\n",*(int*)id,i,l++);
  }
  pthread_exit(id);
}

main()
{
  int h;
  pthread_t hilos[NUM_HILOS];
  int id[NUM_HILOS] = {1,2,3,4,5};
  int error;
  int *salida;
```

```
  for(h=0;h < NUM_HILOS; h++){
    error = pthread_create(&hilos[h],NULL,funcion_hilo,&id[h]);
    if(error){
      fprintf(stderr,"Error %d: %s\n",error, strerror(error));
      exit(-1);
    }
  }

  for(h=0;h<NUM_HILOS;h++){
    error = pthread_join(hilos[h],(void **)&salida);
    if(error)
      fprintf(stderr,"Error %d: %s\n", error, strerror(error));
    else
      printf("Hilo %d terminado\n",*salida);
  }
}
```

# *Ejemplo 1*

- ✓ Si cambio el valor de *i* en un hilo, ¿cambia para el resto?
- ✓ Si cambio el valor de *j* en un hilo, ¿cambia para el resto?
-  Comparar con el último ejemplo de la segunda semana

# *Problemas en la gestión de procesos: necesidad de hilos*

- ✓ Es difícil compartir información entre procesos
- ✓ La generación de procesos mediante *fork*() es cara computacionalmente

```c
#include <stdio.h>
#include <pthread.h>
#include <string.h>

void * mi_hilo(void *arg) {
  printf("my id is %ld\n", (long)pthread_self() );
}

main() {
  pthread_t tid;
  int error;
  error = pthread_create(&tid, NULL, mi_hilo, NULL);
  if (error)
    fprintf(stderr,"Error %d: %s\n", error, strerror(error));
  /*error = pthread_join(pthread_self(),NULL);
  if (error)
  fprintf(stderr,"Error %d: %s\n", error, strerror(error));*/

  error = pthread_join(tid,NULL);
  if (error)
    fprintf(stderr,"Error %d: %s\n", error, strerror(error));

  printf("main thread id %ld, new thread id %ld.\n", (long)pthread_self()
      , (long)tid);

}
```

¿Qué pasa al quitar el comentario que hay en la función main?

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <pthread.h>
void *slowprintf (void *arg) {
  char *msg;
  int i;
  msg = (char *)arg;
  for( i = 0 ; i < strlen(msg) ; i++ ) {
    printf(" %c", msg[i]);
    fflush(stdout);
    usleep (1000000) ;
  }
  pthread_exit(NULL);
}
int main(int argc , char *argv[]) {
  pthread_t h1;
  pthread_t h2;
  char *hola = "Hola ";
  char *mundo = "Mundo";
  pthread_create(&h1, NULL , slowprintf , (void *)hola);
  pthread_create(&h2, NULL , slowprintf , (void *)mundo);
  pthread_join(h1,NULL);
  pthread_join(h2,NULL);
  printf("El programa %s termino correctamente\n", argv[0]);
  exit(EXIT_SUCCESS);
}
```

# *Referencias*


- Francisco M. Márquez. Unix, Programación Avanzada. Editorial: Ra-Ma. 3ª Edición. ISBN: 84-7897-603-5