

Benemérita Universidad Autónoma de Puebla
Facultad de Ciencias de la Computación
Sistemas Operativos 1



“Manejo de procesos”

Pedraza Celón Ian Yael

07/02/2020

Un proceso es una instancia de un programa en ejecución con datos asociados (variables y buffers) y contexto de ejecución, en donde se almacena toda la información que el CPU necesita para ejecutarlo como, por ejemplo: registros, prioridades y eventos por los que espera.

Un proceso tiene un ciclo de vida que consta de cinco estados:

New: Se ejecuta cuando el proceso es creado.

Ready: El estado ready nos indica cuando el proceso está listo para ser ejecutado, este puede ser llamado después de esperar algún otro recurso, como datos de entradas o algún evento.

Running: El proceso entra en estado running cuando se está ejecutando el proceso, desde este estado podemos indicar que necesitamos algún recurso o estamos a la espera de algún evento, como por ejemplo entrada y salida de datos.

Blocked: Este estado bloquea el proceso, lo mantiene en pausa o espera hasta que cierto evento que espere sea ejecutado. Una vez llegado este evento, vuelve a estar en estado *Ready*.

Exit: Es ejecutado cuando el proceso termina y nos regresa un código de salida dependiendo si el proceso fue exitoso o no.

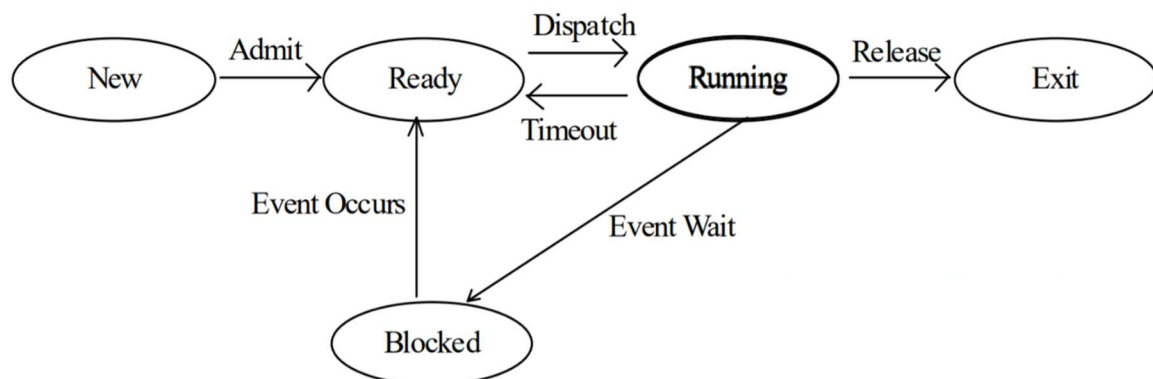


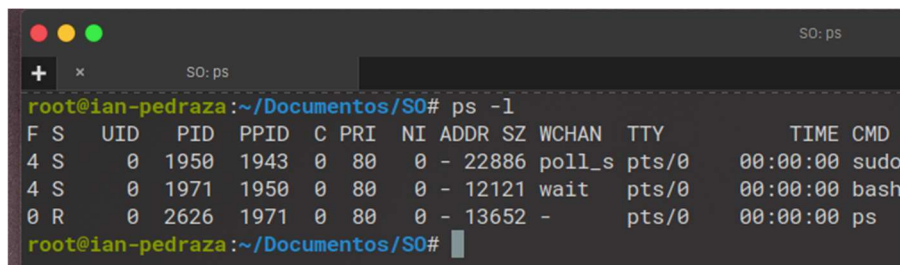
Ilustración 1: Modelo de cinco estados de un proceso

El PCB (*Process Control Block*) es una estructura que contiene información relevante del proceso:

- Identificación (PID, UID, ID proceso padre)
- Información del procesador (PC, CPU registros..., PSW, SP)
- Información de control (Prioridad, estado del proceso, evento por los que espera, links a otros procesos)
- Información de memoria (localización y estado de acceso de los datos usuarios)
- Información de Archivos (archivos y dispositivos abiertos)

Desarrollo:

1. Identifica los procesos que tienes en el sistema utiliza el comando **\$ps**



```
root@ian-pedraza:~/Documentos/S0# ps -l
F S  UID  PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S   0   1950  1943   0  80   0 - 22886 poll_s pts/0      00:00:00 sudo
4 S   0   1971  1950   0  80   0 - 12121 wait   pts/0      00:00:00 bash
0 R   0   2626  1971   0  80   0 - 13652 -      pts/0      00:00:00 ps
root@ian-pedraza:~/Documentos/S0#
```

Ilustración 2: Ejecución del comando \$ps

2. Codifica los siguientes programas y contesta las preguntas:

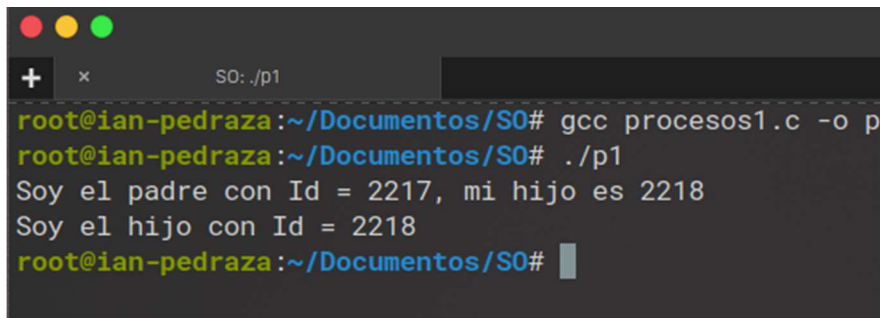
```
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>

int main(){
    pid_t childPid;
    childPid = fork();

    if(childPid == 0){
        printf("Soy el hijo con Id = %ld\n", (long)getpid());
    }else{
        printf("Soy el padre con Id = %ld, mi hijo es %d\n", (long)getpid(), childPid);
    }

    return 1;
}
```

Ilustración 3: Programa 1



```
root@ian-pedraza:~/Documentos/S0# gcc procesos1.c -o p
root@ian-pedraza:~/Documentos/S0# ./p1
Soy el padre con Id = 2217, mi hijo es 2218
Soy el hijo con Id = 2218
root@ian-pedraza:~/Documentos/S0#
```

Ilustración 4: Ejecución del programa 1

Responde las siguientes preguntas:

1. ¿Qué hace el programa?

Crea un proceso, y en caso de ser exitoso muestra el id del proceso hijo, y en caso de no serlo, muestra los datos del proceso y el id del padre.

2. ¿Qué realiza la función getpid()?

Obtiene el id del proceso hijo

3. ¿Por cuales estados pasan los procesos?

New, Ready, Running y Exit

```

#include <stdlib.h>
#include <stdio.h>
#include <unistd.h>

int main(){

    pid_t childPid;
    childPid = fork();

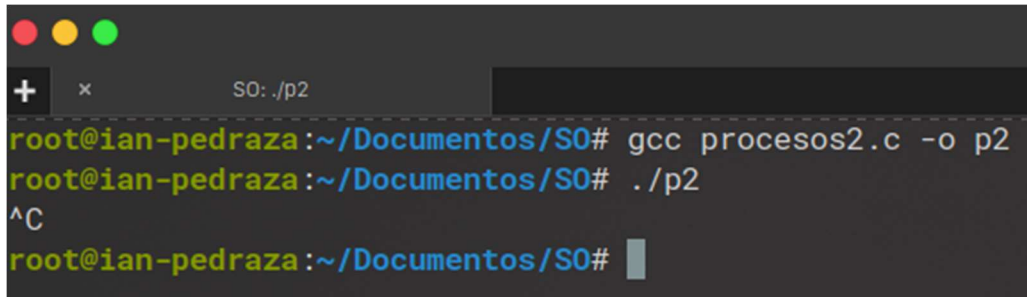
    if(childPid != 0)
        while(1)
            sleep(1000);

    else
        exit(2);

    return 1;
}

```

Ilustración 5: Programa II



The screenshot shows a terminal window with a dark background. The prompt is 'root@ian-pedraza:~/Documentos/S0#'. The user enters 'gcc procesos2.c -o p2' and presses enter. The prompt changes to 'root@ian-pedraza:~/Documentos/S0#'. The user enters './p2' and presses enter. The prompt changes to 'root@ian-pedraza:~/Documentos/S0#'. The user presses Ctrl+C, which is shown as '^C' on the line. The prompt returns to 'root@ian-pedraza:~/Documentos/S0#'.

```

root@ian-pedraza:~/Documentos/S0# gcc procesos2.c -o p2
root@ian-pedraza:~/Documentos/S0# ./p2
^C
root@ian-pedraza:~/Documentos/S0#

```

Ilustración 6: Ejecución del programa II

Responda las siguientes preguntas:

1. ¿Qué hace el programa?

Crea un proceso hijo, y en caso de ser exitoso. Lo duerme por un tiempo indeterminado.

2. ¿Qué función tiene la librería sys/types.h?

Nos proporcionan ciertos objetos para poder manejar los procesos.

- **Determine el estado de los procesos**

Estado: Blocked

- **Mate al proceso padre:** kill -9 -PID
- **Determine el nuevo estado de los procesos**

Estado: Exit

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>

int main(int argc, char *argv[]){
    int i, num_procesos;
    pid_t hijoPid;

    if(argc != 2){
        fprintf(stderr, "Uso: %s num_procesos\n",
argv[0]);
        exit(1);
    }

    num_procesos = atoi(argv[1]);
    hijoPid = 0;

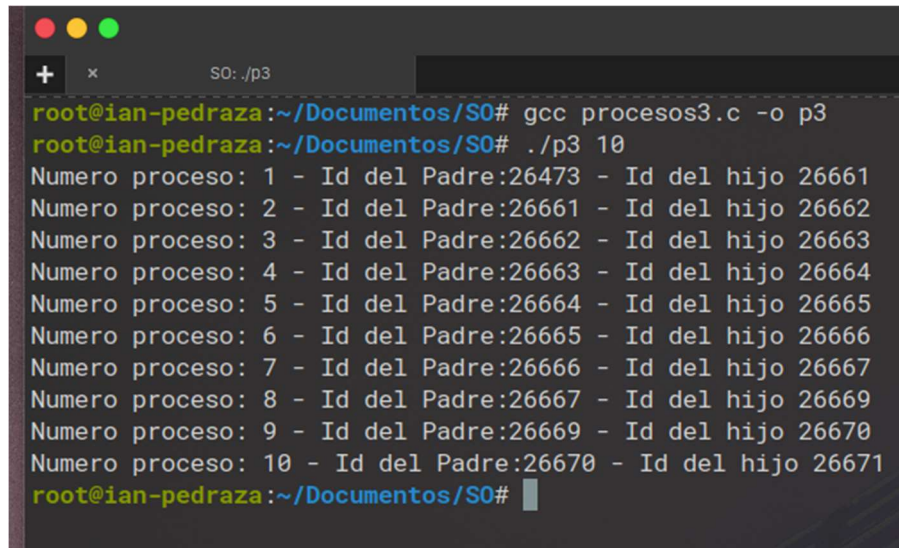
    for(i = 1; i<num_procesos; i++){
        if(hijoPid = fork()){
            break;
        }
    }

    if(hijoPid == -1){
        perror("error al ejecutar el fork");
        exit(1);
    }

    fprintf(stdout, "Numero proceso: %d - Id del
Padre:%ld - Id del hijo %ld\n", i, (long)getppid(),
(long)getpid());
    exit(0);

    return 1;
}
```

Ilustración 7: Programa III



```
root@ian-pedraza:~/Documentos/S0# gcc procesos3.c -o p3
root@ian-pedraza:~/Documentos/S0# ./p3 10
Numero proceso: 1 - Id del Padre:26473 - Id del hijo 26661
Numero proceso: 2 - Id del Padre:26661 - Id del hijo 26662
Numero proceso: 3 - Id del Padre:26662 - Id del hijo 26663
Numero proceso: 4 - Id del Padre:26663 - Id del hijo 26664
Numero proceso: 5 - Id del Padre:26664 - Id del hijo 26665
Numero proceso: 6 - Id del Padre:26665 - Id del hijo 26666
Numero proceso: 7 - Id del Padre:26666 - Id del hijo 26667
Numero proceso: 8 - Id del Padre:26667 - Id del hijo 26669
Numero proceso: 9 - Id del Padre:26669 - Id del hijo 26670
Numero proceso: 10 - Id del Padre:26670 - Id del hijo 26671
root@ian-pedraza:~/Documentos/S0#
```

Ilustración 8: Ejecución del Programa III

Responda las siguientes preguntas:

Ejecute el programa y observe los resultados para diferentes números de procesos.

¿Cuántos de ellos son adoptados?

Uno de ellos es adoptado, el primero

Conclusiones:

Un sistema operativo es el encargado de proveer de servicios a las aplicaciones y administrar los recursos del sistema, y un servicio no es otra cosa que un proceso, y para nosotros como desarrolladores, es muy importante conocer el como se crean y como se manejan.

También es de mucha importancia conocer el ciclo de vida de un proceso, para poder saber como administrar las funciones que ejecuta y en que momento deben realizarse.

Bibliografía.

Bioinformatics at COMAV. (2019). Procesos. 06/02/2020, de COMAV Sitio web:
https://bioinf.comav.upv.es/courses/unix/control_procesos.html

González S.. (2019). Manual básico de administración de procesos . 06/02/2020, de
Linux Total Sitio web:
https://www.linuxtotal.com.mx/index.php?cont=info_admon_012