Práctica 1 (2 ptos)

Simulador de sistema M/M/1

1.- Objetivo

En esta práctica se pretende crear un programa para simular el funcionamiento de un servidor cuyos tiempos de servicio siguen una distribución exponencial. Para completar el sistema se dispone de otro programa que genera llegadas siguiendo un proceso de Poisson.

2.- Herramientas

Se han preparado los siguientes programas para la comprobación del correcto funcionamiento del simulador creado:

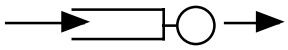
```
LLEGADAS(1)                                          LLEGADAS(1)

NAME
    llegadas

SYNOPSIS
    llegadas tiempo_simulacion tasa_llegadas

DESCRIPTION

    Este programa escribe por la salida estándar paquetes
del formato que espera recibir el programa sim1 de forma
que el campo tstampin forme un proceso de llegadas de
Poisson de tasa tasa_llegadas. Termina cuando una llegada
alcanza el instante tiempo_simulacion.


VUELCA(1)                                              VUELCA(1)

NAME
    vuelca

SYNOPSIS
    vuelca

DESCRIPTION

    Este programa espera recibir por la entrada estándar
paquetes del formato que debe escribir sim1. Se limita a
```

volcar por la salida estándar una línea por cada paquete recibido con el siguiente formato:

```
tstampin  tstampout tserv encola
```

El programa termina al cerrársele la entrada estándar.

3.- Sistema



Las llegadas formarán un proceso de Poisson y los tiempos de servicio deben seguir una distribución exponencial.

La página del manual que podría servir para describir este simulador es la siguiente:

```
SIM1(1)                                                    SIM1(1)

NAME
   sim1

SYNOPSIS
   sim1 tasa_servicios

DESCRIPTION

   El parámetro es la tasa media de servicios por unidad de tiempo.
   El programa debe leer de la entrada estándar estructuras del siguiente tipo:

   #define   MAX_DATA  1500

   typedef struct paquete
   {
        double tstampin; /* Instante de llegada */
        double tstampout; /* Instante de salida */
        double tserv; /* Tiempo de servicio */
        int encola; /* Paquetes en cola a su salida */
        int  tam;
        char data[MAX_DATA];
   } pq;

   Donde el campo tstampin de cada paquete vendrá con la marca de tiempo del instante en el que dicho paquete entra en la cola del sistema (todas las llegadas se producen ordenadas por este campo).
   Tras procesar el paquete el programa debe escribir por la salida estándar el paquete sin alterar los campos tstampin, tam y data[] y colocando en el campo tstampout el
```

instante en el que el paquete termina de ser servido, en el campo tserv su tiempo de servicio y en el campo encola el número de paquetes que quedan en el sistema al producirse la salida de éste.

sim1 produce la salida de un paquete en cuanto tiene toda la información necesaria para rellenar sus campos.

sim1 gestiona dinámicamente la memoria para controlar una cola teóricamente infinita. sim1 puede almacenar en memoria tantos paquetes como necesite y le permita la máquina en que se esté ejecutando pero si no precisa la memoria (por ejemplo los paquetes ya han salido por STDOUT_FILENO o la cola a penas crece) la cede al sistema, minimizando los recursos que emplea.

4.- Ficheros

En el directorio `$(HOME)/solucion/prac1` debe encontrarse un `Makefile` así como todos los ficheros `.c` y `.h` necesarios para crear el programa mencionado en esta práctica. La acción por defecto del `Makefile` (la cual debe funcionar con solo hacer `make` en ese directorio) debe ser crear `sim1`.

Para la corrección de la práctica se borrarán todos los ejecutables, se hará un touch a todos los ficheros fuente y se recompilará mediante el `Makefile`.

En el directorio `$(HOME)/../ficheros/prac1` pueden encontrarse los programas `llegadas` y `vuelca`.

5.- Memoria

Se debe entregar una breve memoria (máximo 3 páginas) comprobando el correcto funcionamiento del simulador comparando los resultados que se obtienen con él con los resultados que predice la teoría de colas.

6.- Aplicaciones, funciones y llamadas al sistema útiles

read(2), write(2), rand(3), srand(3), gnuplot(1), soffice, awk(1).