Se ejecuta en primer lugar el servidor en segundo plano:

Clientes <nombrefifo> <num clientes>

```
mimanejador(int señal)
{printf
}
...
signal (SIGPIPE, mimanejador);
abre <nombrefifo>e  y <nombrefifo>s

crea <num clientes> hijos, cada uno de ellos:

    escribe algo en <nombrefifo>e

    lee <pidproxy> de <nombrefifo>s
    abre fifol<pidproxy> para escritura

    MIENTRAS haya caracteres que escribir {
    escribe en fifo.<pidproxy> //cliente generando datos
    }

    // si se escribe en un fifo aún no abierto para lectura se
    //genera la señal SIGPIPE  que por omisión termina el proceso

while (wait (&estado)!=-1);
```

Servidor <nombrefifo> &

```
mimanejador (int señal)
{pid=wait(&estado);}
.......

signal (SIGCHLD, mimanejador);
.......
crea <nombrefifo>e  y <nombrefifo>s
abre <nombrefifo>e  y <nombrefifo>s para lect. y esc.
crear archivo "bloqueo" que usan los proxys

MIENTRAS NO FIN DE LECTURA {
lee algo de <nombrefifo>e   // ha llegado un cliente nuevo

lanza hijo que será el proxy para este cliente
}
```

```
escribe <pidproxy> en <nombrefifo>s
crea fifo.<pidproxy>


abre fifo.<pidproxy> para lectura (1)

duplica fifo.<pidproxy> en 0

execlp("./proxy", "proxy", 0)
```

```
//no sabe cuántos clientes hay
//concebido para esperar periodos de tiempo que
// podrian ser largos pues un cliente podría demorarse
//un tiempo largo en generar datos
```

¿cuando se borra fifo.<pidproxy>?

proxy

```
MIENTRAS NO FIN DE LECTURA {
lee de 0 y escribe en temporal //temporal es solo de este proceso
}
dbloqueo= open( "bloqueo", O_RWR|O_CREAT)
bloquear (dbloqueo, F_WRLCK)
// ya tenemos la pantalla en ex. mutua
lee de temporal y escribe en 1 // escribimos en pantalla
bloquear (dbloqueo, F_UNLCK)

exit (0)
```