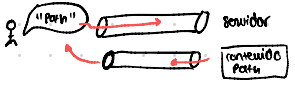
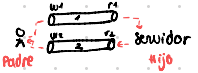
PIPES

OBJETIVO

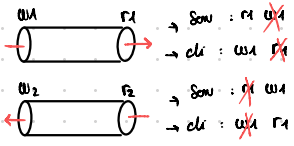


Implementación:



close(r1)
close(w2)
# 1º PIPE
w1.write(path)
close(w1)
# 2º PIPE
r2.read()
close(r2)

close(w1)
close(r2)
# 1º PIPE
r1.read() → path
2º PIPE
w2.write() → contenido
close(w2)




```python
import os
import sys

def cliente_servidor():

    rd1, wd1 = os.pipe() # Tuberia cliente --> serv
    rd2, wd2 = os.pipe() # tuberia serv --> cliente

    r1, w1 = os.fdopen(rd1, 'rb'), os.fdopen(wd1, 'wb')  # cliente-servidor
    r2, w2 = os.fdopen(rd2, 'rb'), os.fdopen(wd2, 'wb')  # servidor-cliente
    pid = os.fork()

    if pid == 0:  #  hijo (servidor)

        w1.close() #os.close(w1)
        r2.close() #os.close(r2)


        path = r1.read().decode()
        print("Serv: he leido archivo del primer pipe")

        #### segunda pipe  ###

        with open(path, 'rb') as file:
            content = file.read()

        print("Serv: he leido el contenido del path")
        w2.write(content) #os.write(w2, contenido)

        print("Serv: he escrito en el pipe2")

        w2.close()
        sys.exit(0)
    else:  #  padre (cliente)

        r1.close() # os.close(r1)
        w2.close() # os.close(w2)

        path_archivo ="/var/log/dpkg.log"
        w1.write(path_archivo.encode())  # os.write(w1, path_archivo.encode())
        w1.close()
        print("Cliente: he escrito el archivo en el pipe")

        ####### segunda pipe ######

        contenido = r2.read()
        print("Cliente: he leido archivo del segundo pipe")
        sys.stdout.write(contenido.decode())

        r2.close() # os.close(r2)

        sys.exit(0)

if __name__ == "__main__":
    cliente_servidor()
```

Abrimos descriptores de ficheros

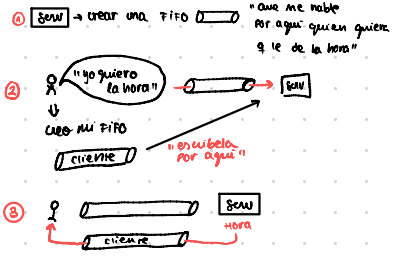
se puede leer como un fichero

FIFO

Objetivo

- comunicación entre cualquier (P)
- Cada (P) puede abrir la FIFO en modo ξ^{R}_{W}
- Son persistentes
- Half-duplex !!

Implementación



servidor.py

```
crear_fifo()
While True:
    read (fifo_Sew) → Path FIFO_Cli
    break
write_fifo_cliente (hora)
```

cliente.py

```
FIFO_Cli = crear_fifo()
Write. fifo1 (Path fifo_cliente)

Leer. fifo_cliente
```

servidor

```python
import os, time, datetime, sys

def calcular_fecha():
    fecha_hora_actual = datetime.datetime.now()

    cadena_fecha_hora = fecha_hora_actual.strftime("%Y-%m-%dT%H:%M:%S%z")
    return cadena_fecha_hora

def crear_fifo(path_fifo):
    if not os.path.exists(path_fifo):
        os.mkfifo(path_fifo)

def main():
    path = "/tmp/fifo_servidor"

    crear_fifo(path)
    print("La FIFO del servidor está en "+ path)

    while True:
        with open(path, 'r') as fifo1:  # abrir fifo1 modo lectura
            while True:
                path_fifo2 = fifo1.read()
                break

            with open(path_fifo2, 'w') as fifo2:
                fifo2.write(calcular_fecha() + "\n")
                fifo2.flush()

if __name__ == "__main__":
    main()
```

cliente

```python
import os, sys

def crear_fifo(path_fifo):
    if not os.path.exists(path_fifo):
        os.mkfifo(path_fifo)

def main():
    path = input("Indica la dirección del servidor: ")
    path_fifo2 = '/tmp/fifo2'
    with open(path, 'w') as fifo1:
        crear_fifo(path_fifo2)
        fifo1.write(path_fifo2)
        fifo1.flush()

    with open(path_fifo2, 'r') as fifo2:
        contenido =  fifo2.read()
        sys.stdout.write(contenido)

        os.remove(path_fifo2)

if __name__ == "__main__":
    main()
```

Fifo servidor: fifo1
Fifo cliente: fifo2