

Systemy operacyjne

Sprawozdanie - laboratorium 7 „Komunikacja międzyprocesowa 2”

Andrzej Kołakowski
296586

1) Łączy nazwane w powłoce

Na poprzednim laboratorium poznaliśmy mechanizm komunikacji między procesami za pomocą łączy nienazwanych, tzw. potoków (ang. pipe).

Łączy nazwane FIFO jest podobne do potoku, jednak jest tworzone w inny sposób. Nie jest anonimowym kanałem komunikacji, ale plikiem specjalnym dodawanym do systemu plików poprzez wywołanie `mkfifo`. Dzięki temu może być wykorzystane do komunikacji między dwoma procesami nieposiadającymi wspólnego przodka.

Aby wyświetlić same katalogi w bieżącym folderze można zastosować następujący filtr:

```
root@localhost:~/Desktop/so
File Edit View Search Terminal Help
[root@localhost so]# ls -l | grep "^d"
drwxr-xr-x. 2 root root 4096 Mar 19 00:47 lab1
drwxr-xr-x. 3 root root 4096 Mar 21 19:10 lab2
drwxr-xr-x. 3 root root 4096 Mar 23 20:56 lab3
drwxr-xr-x. 2 root root 4096 Apr  3 19:55 lab4
drwxr-xr-x. 2 root root 4096 Apr 12 17:29 lab5
drwxr-xr-x. 2 root root 4096 Apr 22 21:09 lab6
drwxr-xr-x. 2 root root 4096 Apr 29 19:49 lab7
[root@localhost so]#
```

Analogiczny efekt można uzyskać za pomocą łączy nazwanych. Otwieramy dwie powłoki, aby nowotworzone procesy nie miały wspólnego przodka. Dzięki temu wykluczemy możliwość dziedziczenia deskryptorów.

W jednej z nich tworzymy kolejkę FIFO o dowolnej nazwie i uruchamiamy `ls -l` przekierowując jego standardowe wyjście na utworzoną kolejkę. W drugiej uruchamiamy `grep "^d"` przekierowując jego standardowe wejście na utworzoną kolejkę. Otrzymujemy analogiczny efekt.

```
root@localhost:~/Desktop/so
File Edit View Search Terminal Help
[root@localhost so]# mkfifo kolejka
[root@localhost so]# ls -l > kolejka
[root@localhost so]#
```

```
root@localhost:~/Desktop/so
File Edit View Search Terminal Help
[root@localhost so]# grep "^d" < kolejka
drwxr-xr-x. 2 root root 4096 Mar 19 00:47 lab1
drwxr-xr-x. 3 root root 4096 Mar 21 19:10 lab2
drwxr-xr-x. 3 root root 4096 Mar 23 20:56 lab3
drwxr-xr-x. 2 root root 4096 Apr  3 19:55 lab4
drwxr-xr-x. 2 root root 4096 Apr 12 17:29 lab5
drwxr-xr-x. 2 root root 4096 Apr 22 21:09 lab6
drwxr-xr-x. 2 root root 4096 Apr 29 19:49 lab7
[root@localhost so]#
```

2) Łączy nazwane w API

Rozważane programy implementują 3 schematy komunikacji klient-serwer między procesami wykorzystując łączy nazwane.

1. Programy wykorzystujące kolejki FIFO

Wprowadzone modyfikacje:

- Serwer po uruchomieniu nie zatrzymuje się na otwieraniu łączy czytania danych od klienta, tylko przechodzi do oczekiwania na dane od klienta.
- Klient po wysłaniu komunikatu nie zatrzymuje się na otwieraniu łączy czytania danych od serwera, tylko przechodzi do oczekiwania na dane od serwera.

Domyślnie otwarcie kolejki FIFO w funkcji `open()` powoduje zatrzymanie wykonywania wątku do czasu aż nie zostanie otwarty przeciwległy koniec. Aby temu zapobiec należy dodać flagę `O_NONBLOCK`. Wtedy otwarcie do odczytu natychmiastowo zakończy się sukcesem, nawet jeżeli przeciwległy koniec nie został jeszcze otwarty. W funkcji `read()` chcemy jednak zatrzymać wykonanie wątku do czasu pojawienia się nadawcy, więc konieczne było usunięcie dodanej flagi. Dodatkowo należało zasymulować odbiorcę na drugim końcu – w przeciwnym wypadku `read()` natychmiast zwróciłoby **EOF** (w rozumieniu `read()` nadawca właśnie zamknął FIFO).

```
root@localhost:~/Desktop/so/lab7
File Edit View Search Terminal Help
[root@localhost lab7]# ./srvfifob
Server started...
Creating server fifo queue 'fifo_root'...OK
Opening server fifo queue 'fifo_root' for reading...
```

Serwer przed zmianami

```
root@localhost:~/Desktop/so/lab7
File Edit View Search Terminal Help
[root@localhost lab7]# ./srvfifo
Server started...
Creating server fifo queue 'fifo_root'...OK
Opening server fifo queue 'fifo_root' for reading...OK
Waiting for data...
```

Serwer po zmianach

```
root@localhost:~/Desktop/so/lab7
File Edit View Search Terminal Help
[root@localhost lab7]# ./cntfifob
Client [5661] started...
Creating client fifo queue 'fifo5661'...OK
Opening server fifo queue 'fifo_root' for writing...OK
Send message: hello world
Writing message to server...OK
Opening client fifo queue 'fifo5661' for reading...
```

Klient przed zmianami

```
root@localhost:~/Desktop/so/lab7
File Edit View Search Terminal Help
[root@localhost lab7]# ./cntfifo
Client [5696] started...
Creating client fifo queue 'fifo5696'...OK
Opening server fifo queue 'fifo_root' for writing...OK
Send message: hello world
Writing message to server...OK
Opening client fifo queue 'fifo5696' for reading...OK
Waiting for data...
```

Klient po zmianach

2. Programy wykorzystujące interfejs z System V

```
root@localhost:~/Desktop/so/lab7
File Edit View Search Terminal Help
[root@localhost lab7]# ./srvsv
Server started...
Creating server key file:
    Path: /tmp/srvsv_root
    Opening server key file...OK
Getting queue key...OK
Getting server queue identifier...OK
Waiting for data...OK
Message from client [98307]: hello world 2
Your response: hello you
Writing response to client 98307...OK
Waiting for data...█
```

Serwer

```
root@localhost:~/Desktop/so/lab7
File Edit View Search Terminal Help
[root@localhost lab7]# ./cntsv
Client started...
Creating server key file name:
    Path: /tmp/srvsv_root
Getting server queue key...OK
Getting server queue identifier...OK
Getting client queue identifier...OK
Your message: hello world 2
Writing response to server...OK
Waiting for data...OK
Message from server: hello you
Your message: █
```

Klient

3. Programy wykorzystujące standard POSIX

Program zawierał błąd, a naszym zadaniem była jego identyfikacja. Po sprawdzeniu otrzymywanego `errno` okazuje się że dostajemy błąd `EMSGSIZE`, który oznacza że rozmiar bufora wiadomości jest mniejszy niż ten zawarty w atrybucie `mq_msgsize` utworzonej kolejki wiadomości. Tworząc kolejkę w funkcji `mq_open` jako `attr` podaliśmy `NULL`, co oznacza, że został użyty domyślny maksymalny rozmiar komunikatu. Można go odczytać za pomocą: `cat /proc/sys/fs/mqueue/msgsize_default`. U mnie wynosi on 8192 bajty. Zatem nasz bufor musi mieć rozmiar co najmniej 8192 bajtów. Po zmianie `MESSAGE_BUF_SIZE` w `psx.h` z 100 na 8192 i podaniu tej stałej do funkcji `mq_send` i `mq_receive` program zaczął działać poprawnie.

Maksymalną ilość wiadomości w kolejce można sprawdzić poleceniem: `cat /proc/sys/fs/mqueue/msg_default`. U mnie wynosi ona 10.

```
root@localhost:~/Desktop/so/lab7
File Edit View Search Terminal Help
[root@localhost lab7]# ./srvpsxb
Server started...
Opening server queue '/psxsrv_psxsrvqueue_root' for reading...OK
Waiting for data...
FAIL!
Error: Message too long
[root@localhost lab7]#
```

Otrzymywany błąd

```
root@localhost:~/Desktop/so/lab7
File Edit View Search Terminal Help
[root@localhost lab7]# ./srvpsx
Server started...
Opening server queue '/psxsrv_psxsrvqueue_root' for reading...OK
Waiting for data...
Message from [6514]: hello world 3
Your response: hi you
Opening client queue '/psxcnt_6514' for writing...OK
Writing response to client 6514...OK
Waiting for data...
█
```

Serwer po zmianach

```
root@localhost:~/Desktop/so/lab7
File Edit View Search Terminal Help
[root@localhost lab7]# ./cntpsx
Server started...
Opening server queue '/psxsrv_psxsrvqueue_root' for writing...OK
Opening client queue '/psxcnt_6514' for reading...OK
Your message: hello world 3
Writing message...OK
Waiting for data...
Message from server: hi you
Your message: █
```

Klient po zmianach

3) Zadania

1. Chatbot

Zadanie polegało na modyfikacji chatbota Eliza tak, aby działał w trybie klient-serwer wykorzystując kolejki FIFO.

```
root@localhost:~/Desktop/so/lab7
File Edit View Search Terminal Help
Message from client [6666]: beautiful
Auto-generated response: I'm sure you like her,don't you ?

Opening client fifo 'fifo6666' for writing...OK
Writing response to client [6666]...OK
Waiting for data...OK
Message from client [6644]: why are we doing this?
Auto-generated response: Remember,therapy is good for you .

Opening client fifo 'fifo6644' for writing...OK
Writing response to client [6644]...OK
Waiting for data...OK
Message from client [6666]: love mcdonalds
Auto-generated response: Remember,love everthing what you love .

Opening client fifo 'fifo6666' for writing...OK
Writing response to client [6666]...OK
Waiting for data...OK
Message from client [6644]: hate mcdonalds
Auto-generated response: So you hate something -- tell me more .

Opening client fifo 'fifo6644' for writing...OK
Writing response to client [6644]...OK
Waiting for data...OK
Message from client [6644]: bye
Auto-generated response: Your bill will be mailed to you .

Opening client fifo 'fifo6644' for writing...OK
Writing response to client [6644]...OK
Waiting for data...
```

Serwer w trakcie rozmowy z dwoma klientami: 6666 i 6644

```
root@localhost:~/Desktop/so/lab7
File Edit View Search Terminal Help
Send message:
> beautiful
Writing message to server...OK
Opening client fifo queue 'fifo6666' for reading...OK
Waiting for data...OK
Message from server: I'm sure you like her,don't you ?

Send message:
> love mcdonalds
Writing message to server...OK
Opening client fifo queue 'fifo6666' for reading...OK
Waiting for data...OK
Message from server: Remember,love everthing what you love .

Send message:
> 
```

Klient 6666

```
root@localhost:~/Desktop/so/lab7
File Edit View Search Terminal Help
Send message:
> why are we doing this?
Writing message to server...OK
Opening client fifo queue 'fifo6644' for reading...OK
Waiting for data...OK
Message from server: Remember,therapy is good for you .

Send message:
> hate mcdonalds
Writing message to server...OK
Opening client fifo queue 'fifo6644' for reading...OK
Waiting for data...OK
Message from server: So you hate something -- tell me more .

Send message:
> bye
Writing message to server...OK
Opening client fifo queue 'fifo6644' for reading...OK
Waiting for data...OK
Message from server: Your bill will be mailed to you .

[root@localhost lab7]# 
```

Klient 6644

2. Zliczanie głosów

Zadanie polegało na napisaniu aplikacji, która zlicza głosy z poszczególnych okręgów wyborczych posługując się kolejką komunikatów. Do ich przesyłania wybrałem POSIX.

Założenia:

- Okręgi to aplikacje klienckie generujące losowo liczbę głosów
- Po każdorazowym otrzymaniu informacji od okręgu wyborczego, serwer przesyła do każdego zarejestrowanego okręgu aktualne dane o frekwencji wyborczej
- Informacje o liczbie głosów dla poszczególnych komitetów wyborczych są przesyłane poprzez priorytet komunikatu
- Na podstawie otrzymanych informacji serwer wyświetla informacje o frekwencji oraz który komitet wygrał wybory


```
root@localhost:~/Desktop/so/lab7
File Edit View Search Terminal Help
[root@localhost lab7]# ./srv_pkw_2
Server started...
Opening server queue '/psxsrv_psxsrvqueue_root' for reading...OK
Waiting for data...
Message from [6813]: 3
Waiting for data...
Message from [6813]: 2
Waiting for data...
Message from [6813]: 0
Waiting for data...
Message from [6813]: 0

Aktualna frekwencja: 5

Opening client queue '/psxcnt_6813' for writing...OK
Writting responce to client 6813...OK
Waiting for data...
Message from [6814]: 4
Waiting for data...
Message from [6814]: 0
Waiting for data...
Message from [6814]: 1
Waiting for data...
Message from [6814]: 0

Koncowa frekwencja: 10
Komitet nr: 3 glosy: 7
Komitet nr: 2 glosy: 2
Komitet nr: 1 glosy: 1
Komitet nr: 0 glosy: 0
Zwyciezca: komitet nr 3

Opening client queue '/psxcnt_6814' for writing...OK
Writting responce to client 6814...OK
Opening client queue '/psxcnt_6813' for writing...OK
Writting responce to client 6814...OK
[root@localhost lab7]#
```

Serwer odbierający dane z 2 okręgów wyborczych: 6813 i 6814

```
root@localhost:~/Desktop/so/lab7
File Edit View Search Terminal Help
[root@localhost lab7]# ./cnt_pkw_2
Server started...
Opening server queue '/psxsrv_psxsrvqueue_root' for writing...OK
Opening client queue '/psxcnt_6813' for reading...OK
Writting message...

j=3 votes=3
j=2 votes=2
j=1 votes=0
j=0 votes=0

OK
Waiting for data...
Message from server: Aktualna frekwencja: 5

Waiting for data...
Message from server:
Koncowa frekwencja: 10

[root@localhost lab7]#
```

Klient (okręg) 6813

```
root@localhost:~/Desktop/so/lab7
File Edit View Search Terminal Help
[root@localhost lab7]# ./cnt_pkw_2
Server started...
Opening server queue '/psxsrv_psxsrvqueue_root' for writing...OK
Opening client queue '/psxcnt_6814' for reading...OK
Writting message...

j=3 votes=4
j=2 votes=0
j=1 votes=1
j=0 votes=0

OK
Waiting for data...
Message from server:
Koncowa frekwencja: 10

[root@localhost lab7]#
```

Klient (okręg) 6814