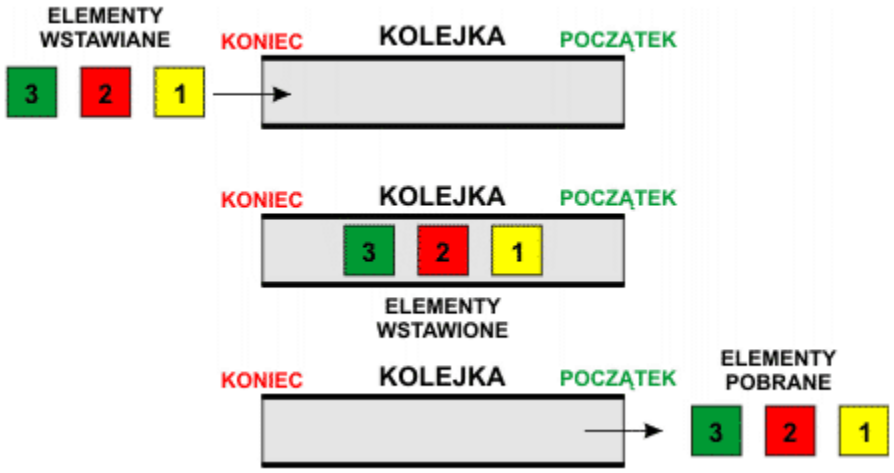
Systemy Operacyjne

ZAAWANSOWANA KOMUNIKACJA MIDZYPROCESOWA
KOLEJKI

Zajęcia 7 – ZAAWANSOWANKOMUNIKACJA MIĘDZYPROCESOWA

- 1 Łącza nazwane
  - 1.1 kolejki FIFO
  - 1.2 System V - IPC
  - 1.1 POSIX IPC
- 2. Zadania
  - 2.1 Chatbot
  - 2.2 Zliczanie głosów

Kolejki w systemie linux



kolejka FIFO

http://eduinf.waw.pl/inf/alg/001_search/0105.php

- Kolejki systemowe można utworzyć:
  - mkfifo test
- Podobnie do plików i pipe
- Przykład (ls –l | grep ala) :
  - (KONSOLA 1) ls –l > test
  - (KONSOLA 2) grep ala < test
- Tworzenie i statystyki

```
abasiura@galaxy:~/SO/7/7_0$ mkfifo kolejka
abasiura@galaxy:~/SO/7/7_0$ ls -l
total 16
prw-r--r-- 1 abasiura eaiibgrp     0 2017-04-24 19:36 kolejka|
-rwxr-xr-x 1 abasiura eaiibgrp 16158 2017-03-13 15:22 sysinfo_3_1*
abasiura@galaxy:~/SO/7/7_0$
abasiura@galaxy:~/SO/7/7_0$ stat kolejka
  File: `kolejka'
  Size: 0               Blocks: 0          IO Block: 4096   fifo
Device: 11h/17d Inode: 116793846   Links: 1
Access: (0644/prw-r--r--)  Uid: ( 7668/abasiura)   Gid: (  434/eaiibgrp)
Access: 2017-04-24 19:36:16.000000000 +0200
Modify: 2017-04-24 19:36:16.000000000 +0200
Change: 2017-04-24 19:36:16.000000000 +0200
abasiura@galaxy:~/SO/7/7_0$
```

Kolejki systemowe

- Funkcje w C
  - mkfifo(const char *path, /* pathname */
    mode_t perms /* permissions */
) - tworzenie kolejki

  - open – po utworzeniu musimy otworzyć kolejkę
  - O_NONBLOCK

- Operacje we/wy na deskryptorach
  - open (oczekiwanie na drugą stronę)
  - write
  - read
  - close
  - fstat
  - lseek (?)
  - unlink

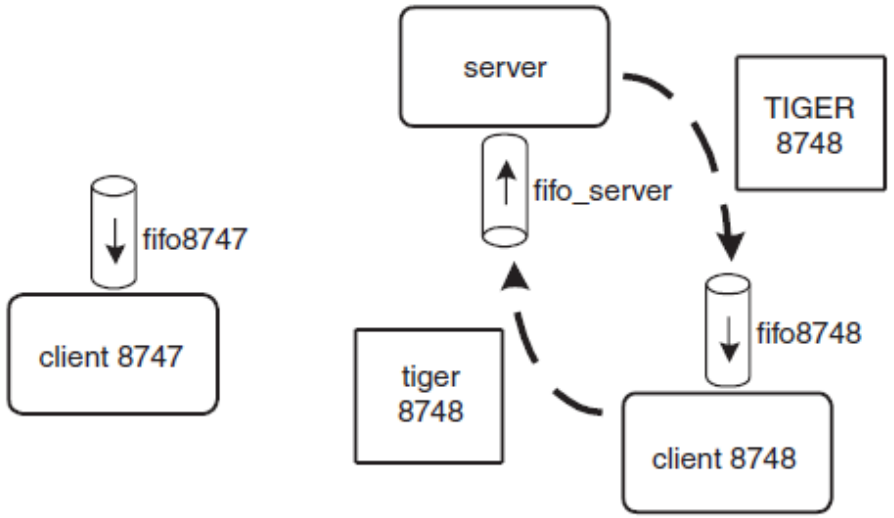


Figure 7.1 Server with two clients.

System-V (podstawowe operacje)

- Mechanizmy IPC
  - msg, sem, shm
  - *X*get, *X*ctl (czyli msgget , msgctl)

- Xget – tworzy objekt , brak deskryptorów - klucze do obiektów systemowych
  - int **msgget**(key_t **key**, int **msgflg**)

  - #define SEM_KEY 0001
  - ftok(path, id) - generuje klucz
  - IPC_PRIVATE (identyfikator 0)
  - IPC_CREAT , IPC_EXCL | S_IRWUSR …

- *Xctl:*
  - int **msgctl**(int **msgid**, int **cmd**, struct msgid_ds ***buf**)
  - IPC_STAT – stan obiektu
  - IPC_SET – zmienia właściciela, grupę i tryb objektu
  - IPC_REM – usuwa obiekt z pamięci

  - Struct **msgid.ipc_perm** – uprawnienia

System-V (kolejki – operacje)

- Podstawowe operacje (linux)
  - ipcs –q/s/m
  - ipcrm –q/s/m

- Wysyłanie
  - int msgsnd(int msgid, struct msgbuf *msgp, int msgs, int msgflg)
  - struct msgp(mtype, mtext)
  - IPC_NOWAIT

- Odbieranie wiadomości
  - int msgrcv (int msgid, struct msgbuf *msgp, int msgs, long msgtyp, int msgflg)
  - msgtyp = typ komunikatu (4, -4, 0). Jakie wartości dostaniemy jeśli podany -4 ?
  - msgflg = 0 | IPC_NOWAIT | MSG_NOERROR

- struct **msg** (long **mtype**; char mtext[MTEXTSIZE]; } - dowolna struktura ale mtype

- Odbiór i zapis - kopiowanie pomiędzy buforami

POSIX

- Tworzenie
  - **mqd_t** mq_open (char *name, int flags, int mode, mq_attr * attr)
  - O_CREAT | O_RDONLY ..
  - struct mq_attr { …, long **mq_maxmsg** , long **mq_msgsize** long **mq_curmsgs** }

- Wysyłanie
  - int mq_send(mqd_t **mgd**, const char* **msg**, size_t **msgsize**, unsigned **priority**)
  - MQ_PRIORITY_MAX = 32

- Odbieranie wiadomości
  - int mq_receive (mqd_t **mgd**, const char* **msg**, size_t **msgsize**, unsigned * **priorityp**)
  - najwyższy priorytet lub najstarsza wiadomość
  - priorityp – priorytet odbieranej wiadomości

- Inne
  - int mq_close(mqd_t mq)
  - int mq_unlink(char *name)

  - int mq_notify(mqd_t mq, struct sigevent *notif) – po odebraniu eventu na nowo należy ustawić
  - int mq_getattr(mqd_t mq, struct mq_attr *attr)

Ćwiczenia

- Łącza nazwane w powłoce
- Łącza nazwane w API

- 1 – chatbot (+)
- 2 – zliczanie głosów (++)

M.J.Rochkind - Programowanie w systemie UNIX dla zaawansowanych (*Advanced UNIX Programming*). Wydanie 3PL / 2EN

Proszę przeczytać rozdział:

- Rozdział 7. Zaawansowana komunikacja międzyprocesowa (>7.7)

Zapoznać się /przypomnieć sobie pojęcia sekcja krytyczna, zakleszczenie (deadlock), zagłodzenie(starvation) w kontekście pracy z procesami i wątkami.