Metody komunikacji międzyprocesowej

Tymon Tobolski
Politechnika Wrocławska
2012

- pliki
- semafory
- łącza (nazwane i nienazwane) (*pipes*)
- kolejki (*queues*)
- pamięć dzielona (*shared memory*)
- gnizda (*sockets*)
- inne

- pliki
- semafory
- łącza (nazwane i nienazwane) (*pipes*)
- kolejki (*queues*)
- pamięć dzielona (*shared memory*)
- gniazda (*sockets*)
- inne

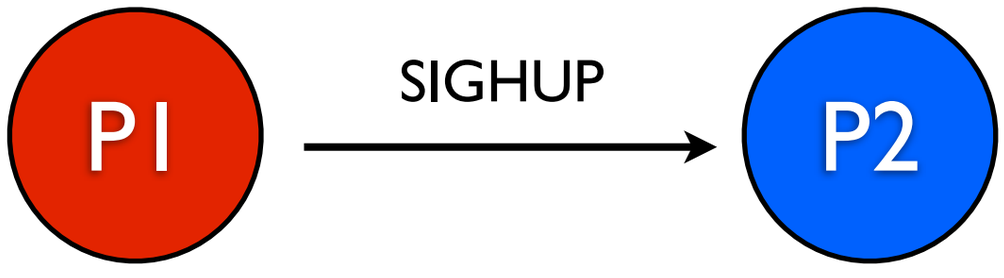
Pliki

- ...

Sygnały

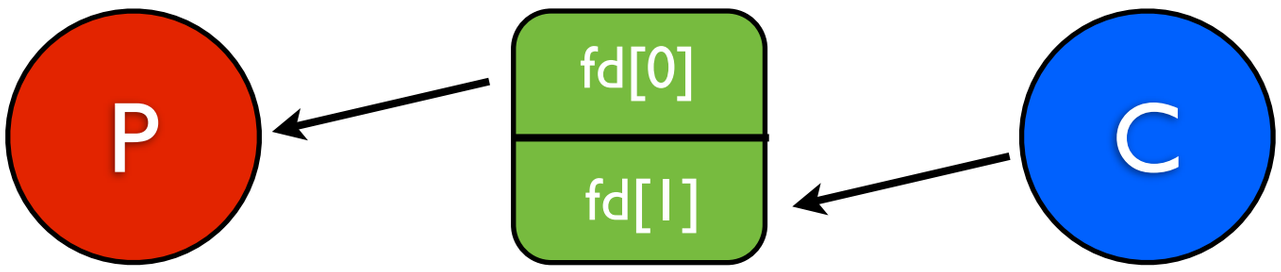
- Proste i ograniczone
- Asynchroniczne
- Przerwanie programu
- `$ kill`

```c
#include <signal.h>

void handler(int signum){
  // ...
}

int main(){
  signal(SIGHUP, handler);
  // ...
}
```



Łącza nienazwane

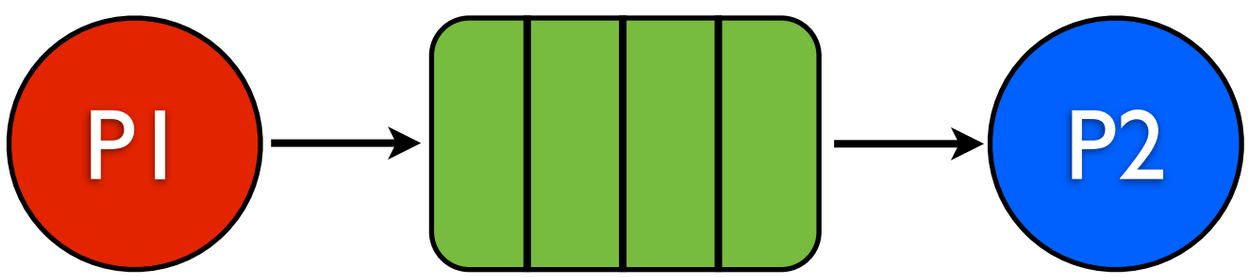
- Tylko dla procesów *parent <> child*
- Domyślnie blokujące




```c
int fd[2];

pipe(fd);

if(fork() == 0){
  close(fd[0]);
  write(fd[1], ...);
  //...
  close(fd[1]);
} else {
  close(fd[1]);
  read(fd[0], ...);
  close(fd[0]);
}
```

Łącza nazwane

- Specjalnie pliki FIFO
- Domyślnie blokujące

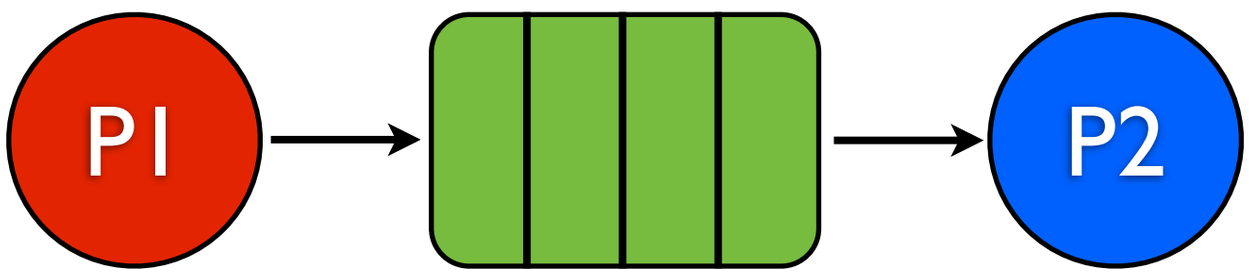



```c
// write
int fd;

mkfifo("rura", O_RDWR |
O_CREAT);

fd = open("rura", O_WRONLY);
write(fd, ...);
close(fd);


// read
int fd;

fd = open("rura", O_RDONLY);
read(fd, ...);
close(fd);
```

Kolejki

- Wiadomości
- Brak zależności czasowych

```
#include <mqueue.h>

mq_open(...);
mq_send(...);
mq_receive(...);
mq_close(...);
mq_unlink(...);
```



Gniazda

- Unix sockets
- TCP / UDP
- Klient - Serwer

```
#include <mqueue.h>

socket(...);
bind(...);
listen(...);
accept(...);
connect(...);
```

Inne

- RPC
- Baza danych
- Pub/Sub