Programowanie współbieżne i rozproszone – laboratorium

Instrukcja do ćwiczenia laboratoryjnego I

Procesy

Autor: S. Samolej

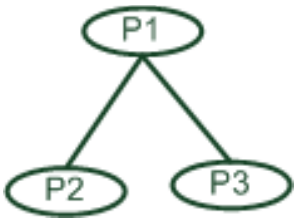
Rzeszów, 2013

Podziękowania:
Składam podziękowania panu dr inż. Jędrzejowi Ułasiewiczowi za udostepnienie materiałów wykładowych i laboratoryjnych, które były inspiracją do opracowania tych materiałów.

1. Proszę uruchomić terminal i przetestować działanie poleceń:
   ps – pobieranie informacji o aktywnych procesach
   ps –ef – wszystkie procesy w systemie
   ps – axjf – drzewo procesów
   ps – sLf - informacja o wątkach

   top – dynamiczna lista aktywnych procesów w systemie
   pstree – drzewo procesów w systemie

   Na podstawie instrukcji obsługi poleceń (np.: man ps) należy dowiedzieć się, jakie informacje zostają podane po wywołaniu podanych poleceń.
2. Proszę skompilować i uruchomić przykładowy program „fork1.c”.
   a. Należy wydłużyć działanie obu procesów z programu i sprawdzić odpowiednimi instrukcjami (ps –ef |grep fork), czy w systemie pracują 2 procesy i jakie są ich parametry.
   b. Proszę zamienić ilość pętli wykonywanych przez procesy i wykryć z zastosowaniem odpowiednich poleceń systemowych, że jeden z procesów jest w stanie zombie.
3. Proszę skompilować i uruchomić przykładowy program „wait.c”.
   a. Należy porównać działanie tego programu z działaniem programu „fork1.c” i zwrócić uwagę na możliwość przechwycenia numeru PID procesu, z którym synchronizuje się proces macierzysty.
4. Proszę zaproponować program, który będzie miał 2 bezpośrednie procesy potomne jak na schemacie:



5. Proszę zaproponować program, który będzie tworzył kaskadę procesów potomnych jak na schemacie:

6. Proszę skompilować i uruchomić przykładowe programy „**pipe1.c**”. Należy zwrócić uwagę w jaki sposób program powołuje do życia nienazwany potok i przesyła sam do siebie przez niego strumień danych.
7. Proszę skompilować i uruchomić przykładowe programy „**pipe2.c**”. Należy zwrócić uwagę w jaki sposób w programie składającym się z 2 procesów zorganizowano komunikację pomiędzy procesami z zastosowaniem nienazwanego potoku.
8. Proszę skompilować i uruchomić program „**fifo1.c**”. Program tworzy systemową kolejkę fifo (potok nazwany) widoczną w systemie plików jako **/tmp/my_fifo**. Należy przetestować działanie kolejki z zastosowaniem następujących poleceń:
   cat < /tmp/fifo1 &
   echo "Ala ma kota" > /tmp/fifo1
9. Proszę skompilować programy „fifo3.c” i „fifo4.c” i zwrócić uwagę, w jaki sposób została zorganizowana w tych dwóch programach komunikacja z zastosowaniem kolejki. Proszę uruchomić programy sekwencją poleceń:
   ./fifo3 &
   time ./fifo4
10. Proszę napisać aplikację składającą się z procesów P1 i P2. Proces P2 jest procesem potomnym procesu P1. Proces P1 przekazuje przez potok nienazwany co 1 sekundę do P2 kolejne liczby 1,2,…,10 które mają być wyświetlane przez P2.
11. Proszę napisać 2 programy. Jeden z nich ma wysyłać do potoku nazwanego łańcuchy tekstowe o stałej długości (np. „tekst 01”, „tekst 02”,…) co sekundę przez 10 sekund. Drugi program ma odbierać takie łańcuchy z tego samego potoku. Kolejki mają być otwarte w trybie blokującym. Proszę przeanalizować działanie systemu złożonego z 2 programów wysyłających i jednego programu odbierającego.