Marcin Cieślikowski

Marek Jasiński

Paweł Goździkowski

Dokumentacja wstępna – projekt UXP1A

1) Interpretacja treści zadania

Należy napisać program zarządcy blokad oraz bibliotekę funkcji, dzięki którym procesy działające na współdzielonych zasobach, będą synchronizowane w dostępie do tych zasobów. Komunikacja i synchronizacja pomiędzy zarządcą blokad, a pozostałymi procesami będzie się odbywać poprzez potoki nazwane (FIFO). Procesy w celu uzyskania dostępu do zasobu będą musiały wywołać funkcję biblioteczne. Procesy mogą zakładać różne typy blokad tj. Concurrent Read, Concurrent Write, Protected Read, Protected Write oraz Exclusive. Proces zarządcy blokad roztrzygał będzie o możliwości założenia blokady przez proces na podstawie aktualnie założonych blokad, zgodnie z macierzą współzależności między typami blokad. W celu uniknięcia zagłodzenia możliwość przydzielenia procesowi zasobu jest rozpatrywana wyłącznie wtedy, gdy inny proces, który wcześniej zgłosił żądanie, nie czeka na ten zasób.

2) Krótki opis funkcjonalny – „black-box"

Proces DLM – przy starcie inicjalizuje potok nazwany (FIFO), który służyć będzie do odbierania komunikatów od procesów. Potok ma stałą i znaną nazwę, oraz ścieżkę dostępu. Proces inicjalizuje struktury przechowujące informację o zasobach. Maksymalna liczba zasobów jest ograniczona. Proces zawiesza się w oczekiwaniu na żądania dostępu do zasobów współdzielonych. Żądania obsługiwane są w kolejności zgłoszeń.

Biblioteka funkcji – składa się z trzech funkcji:

- int DLM_lock(int resource_id, int lock_type, long timeout) – wysyła komunikat do zarządcy blokad. Komunikat składa się z identyfikatora zasobu, typu blokady, czasu timeout, identyfikatora procesu (PID). Następnie tworzony jest potok nazwany (FIFO). Proces zawiesza się w oczekiwaniu na komunikat od zarządcy blokad. Typy komunikatów zwrotnych od procesu zarządcy:

- zasób przydzielony

- minął czas oczekiwania (timeout)

- ponowna próba zajęcia tego samego zasobu

- niewłaściwy identyfikator zasobu

- niewłaściwy typ blokady.

- niewłaściwy czas oczekiwania (timeout)

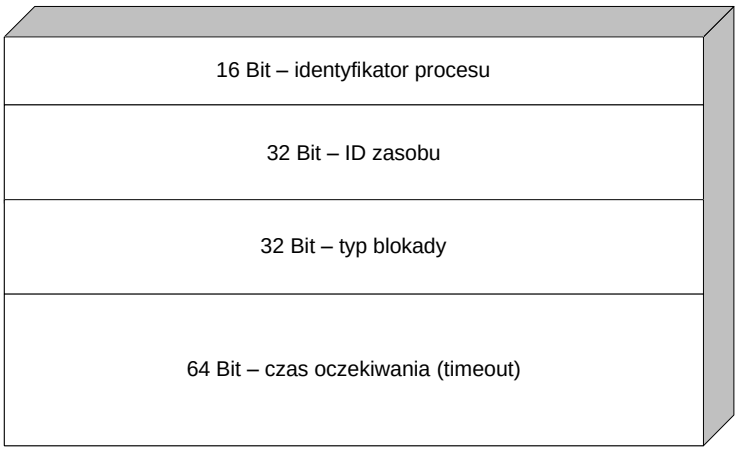
Po odebraniu komunikatu proces usuwa potok.

- int DLM_unlock(int resource_id) – wysyła do DLM komunikat o zwolnieniu zasobu składający się z identyfikatora zasobu oraz identyfikatora procesu (PID).

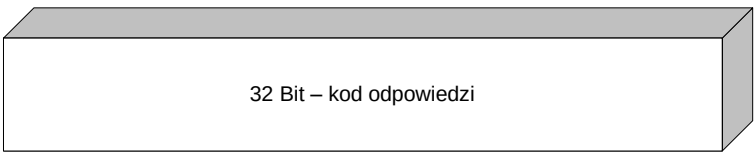
- int DLM_trylock(int resource_id, int lock_type) – wywołuje funkcję lock() z parametrem timeout=-1

3) Opis i analiza poprawności stosowanych protokołów komunikacyjnych

Rysunki przedstawiają strukturę komunikatów przesyłanych do i od DLM. Komunikaty obsługiwane są przez DLM w kolejnośći zgłaszania, co wynika ze struktury potoku FIFO. Proces oczekujący na odpowiedź od DLM zawiesza się na utworzonym potoku w oczekiwaniu na tę odpowiedź. Po odebraniu wiadomości potok odbiorczy procesu jest kasowany, dzięki czemu zakończone procesy nie powodują pozostania nieużywanych potoków. Nazwa potoku odbiorczego procesu jest ściśle ustalona i zawiera numer identyfikacyjny procesu (PID). Aby komunikaty były przesyłane atomowo muszą być mniejsze niż pojemność bufora kolejki FIFO. Pojemność bufora kolejki w Linuxie to 4096B, a standard POSIX 1-2001 wymaga bufora większego od 512B. Komunikaty do i z DLM mają odpowiednio 18B i 4B, dzięki czemu będą przesyłane atomowo.



Rys 1. Komunikat do DLM.



Rys 2. Komunikat odpowiedzi DLM.

4) Planowany podział na moduły i struktura komunikacji między nimi

Rysunek przedstawia schemat podziału na moduły oraz strukturę komunikacji pomiędzy nimi.