Marcin Cieślikowski

Marek Jasiński

Paweł Goździkowski

**Dokumentacja końcowa – 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 rozstrzygał 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) Pełen 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. Jest to /tmp/DLM/DLMfifo. Proces inicjalizuje struktury przechowujące informację o zasobach. Proces zawiesza się w oczekiwaniu na żądania dostępu lub zwolnienia 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) o nazwie odpowiadającej pid procesu. Proces zawiesza się w oczekiwaniu na komunikat od zarządcy blokad. Typy komunikatów zwrotnych od procesu zarządcy:

- zasób przydzielony

- zasób zajęty (dla trybu NONBLOCK, który odpowiada timeout = -1 = NONBLOCK)

- minął czas oczekiwania (timeout)

- błąd otwarcia potoku do wysyłania żądań

-błąd utworzenia potoku do odbierania żądań

-błąd otwarcia potoku do odbierania żądań

-błąd zapisu do potoku

-błąd odczytu z potoku

- ponowna próba zajęcia tego samego zasobu

- niewłaściwy typ blokady.

- niewłaściwy czas oczekiwania (timeout)

Po odebraniu komunikatu proces usuwa potok.

**- int DLM\_unlock(int resource\_id)** – wysyła do DLM żądanie zwolnienia zasobu składające się z identyfikatora zasobu oraz identyfikatora procesu (PID). Jako typ blokady przekazywany jest specjalny parametr FREERESOURCE. Parametr timeout nie ma tutaj znaczenia. Następnie tworzony jest potok nazwany (FIFO) o nazwie odpowiadającej pid procesu. Proces zawiesza się w oczekiwaniu na komunikat od zarządcy blokad. Typy komunikatów zwrotnych od procesu zarządcy:

- zasób zwolniony

- zasób niezwolniony

- błąd otwarcia potoku do wysyłania żądań

-błąd utworzenia potoku do odbierania żądań

-błąd otwarcia potoku do odbierania żądań

-błąd zapisu do potoku

-błąd odczytu z potoku

Po odebraniu komunikatu proces usuwa potok.

**- int DLM\_trylock(int resource\_id, int lock\_type)** – wysyła żądanie, w którym parametr timeout = TRYLOCK, gdzie TRYLOCK oznacza zdefiniowaną stałą. DLM odpowiada na takie zgłoszenie możliwie najszybciej. Typy komunikatów zwrotnych procesu zarządcy:

- zasób wolny

- zasób zajęty

- błąd otwarcia potoku do wysyłania żądań

-błąd utworzenia potoku do odbierania żądań

-błąd otwarcia potoku do odbierania żądań

-błąd zapisu do potoku

-błąd odczytu z potoku

- niewłaściwy typ blokady.

Po odebraniu komunikatu proces usuwa potok.

**3) Opis i analiza poprawności stosowanych protokołów komunikacyjnych**

Rysunki przedstawiają strukturę komunikatów przesyłanych do i od DLM. Komunikaty obsługiwane są przez DLM w kolejności 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) Podział na moduły i struktura komunikacji między nimi**

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

