Dokumentacja wstępna wariant z kolejką komunikatów

Zespół 3:

- Adam Svystun
- Gabriela Ossowska
- Krzysztof Urbanowicz
- Piotr Słysz

1.Treść zadania

Napisać wieloprocesowy system realizujący prosty system (w przestrzeni użytkownika, nie jądra). System powinien udostępniać następujące funkcje (zachowanie analogiczne do systemowych odpowiedników):

```
int simplefs_open(char* name, int mode);
int simplefs_unlink(char* name);
int simplefs_mkdir(char* name);
int simplefs_creat(char* name, int mode);
int simplefs_read(int fd, char* buf, int len);
int simplefs_write(int fd, char* buf, int len);
int simplefs_lseek(int fd, int whence, int offset);
```

System powinien być zrealizowany w jednym pliku, zachowującym się jak uproszczony dysk logiczny, tj. powinien on zawierać: tablicę i-node-ów, katalog główny, listę wolnych bloków, obszar na pliki. Należy zaimplementować tylko pliki zwykłe i katalogi.

Implementacja powinna pozwalać na jednoczesny dostęp wielu procesów, tak więc należy zrealizować wykluczanie i wielodostęp, nie na poziomie całego mini-systemu plików, ale możliwie “drobno-ziarniście”, tj. na poziomie poszczególnych obiektów (katalogów, tablicy i-node-ów). Wykluczanie zrealizować przy pomocy kolejek komunikatów.

2. Interpretacja treści

Stworzyć bibliotekę udostępniającą funkcje umożliwiające stworzenie i obsługę systemu plików realizowanym w pliku traktowanym jako dysk logiczny. System dopuszcza tworzenie hierarchii katalogów przechowujących pliki o ustalonych prawach dostępu read/write, tych samych dla wszystkich procesów. Z biblioteki korzystać będą działające współbieżnie programy testowe próbując tworzyć, modyfikować i usuwać pliki lub katalogi. Synchronizacja dostępu do sekcji krytycznej w korzystaniu z systemu plików jest realizowana przy pomocy kolejek komunikatów - procesy, które chcą dostępu do pliku/katalogu, sprawdzają czy w kolejce jest komunikat o dostępności żądanego zasobu. Jeśli jest to działają na pliku/katalogu, na końcu wysyłając komunikat o dostępności, a jeśli nie ma to czekają aż taki komunikat się pojawi. Zablokowanie katalogu powoduje zablokowanie również wszystkich plików i katalogów, które się w nim znajdują. System korzysta z systemowej biblioteki sys/msg.h do obsługi kolejek.

3. API

`int simplefs_open(char* name, int mode);` - otwiera plik ‘name’ w trybie ‘mode’ i zwraca deskryptor pliku

`int simplefs_close(int fd);` - zamyka plik o podanym deskryptorze

`int simplefs_unlink(char* name);` - usuwa plik ‘name’

`int simplefs_mkdir(char* name);` - tworzy katalog ‘name’

`int simplefs_creat(char* name, int mode);` - tworzy plik ‘name’ z prawami dostępu ‘mode’ i zwraca id pliku

`int simplefs_read(int fd, char* buf, int len);` - czyta len znaków do bufora buf z pliku o id fd

`int simplefs_write(int fd, char* buf, int len);` - pisze len znaków z bufora buf do pliku o id fd

`int simplefs_lseek(int fd, int whence, int offset);` - przesuwa wskaźnik w otwartym pliku o id fd o ‘offset’ bajtów względem flagi ‘whence’

`int simplefs_mount(char* name, int size, int inodes)` - tworzy nowy system plików o danej nazwie albo montuje istniejący

`int simplefs_unmount(char* name)` - usuwa system plików o danej nazwie

4. Opis i analiza poprawności zaproponowanych protokołów komunikacyjnych

Dla komunikacji między procesami używamy kolejek komunikatów. Kolejki komunikatów można najlepiej opisać jako listę kierunkową w przestrzeni adresowej jądra. Wiadomości mogą być wysyłane do kolejki w pewnej kolejności, a potem pobierane z kolejki na kilka różnych sposobów.

Przy pomocy kolejek komunikatów realizujemy wykluczanie i wielodostęp.

Pod czas inicjalizacji naszego systemu plików tworzymy oddzielną kolejkę dla zarządzania dostępami procesów. Dla każdego pliku wrzucamy do kolejki komunikat o dostępności tego pliku, robimy to przez ustawianie typu komunikatu jako numer i-noda związanego z tym plikiem. Kiedy jakiś proces chce wykonać na pliku o deskryptorze fd operację wymagającą uzyskania wyłącznego dostępu do tego pliku, to sprawdza czy jest w kolejce komunikat o typie - fd. Jeśli jest to zaczytuje ten komunikat i po wykonaniu operacji na pliku ponownie wstawia komunikat odpowiadający plikowi do kolejki.

5. Podział na moduły/procesy i struktura komunikacji między nimi

Program składa się z modułów:

- Zarządzania systemem plików odpowiedzialnym za wykonywanie operacji na plikach i katalogach oraz zapewnienie poprawnego działania wykluczania i wielodostępu. Moduł zarządza też wewnętrzną strukturą systemu plików oraz i-nodami.
- Moduł obsługi błędów odpowiada za obsługę sytuacji wyjątkowych i informuje użytkownika o rodzaju błędu.

Procesy w systemie komunikują się przy użyciu kolejki komunikatów.

6. Zarys koncepcji implementacji

Struktury danych:

inodes – tablica zawierająca informacje o plikach

struct inodeFree – struktura przedstawiająca wolny blok i zawierająca wskaźnik na następny wolny blok

Funkcje:

init – tworzy lub otwiera system plików i kolejkę komunikatów

defragment - defragmentacja plików

ls - wyświetlanie zawartości katalogu o podanej ścieżce

update_mem – aktualizuje informację o dostępnej wolnej pamięci