Weryfikacja Synchronizacji

Zuzanna Santorowska

May 2020

Treść Zadania

Należy napisać w środowisku systemu Minix program w jezyku C (oraz skrypt demonstrujacy wykorzystanie tego programu) realizujacy podstawowe funkcje systemu plików.

System plików należy zorganizować w dużym pliku o zadanej wielkości, który bedzie "wirtualnym dyskiem". Program powinien tworzyć dysk wirtualny, oraz dokonywać zapisów i odczytów w celu zrealizowania podstawowych operacji na dysku, zwiazanych z zarzadzaniem katalogiem, alokacja plików oraz utrzymywaniem unikalności nazw.

W pliku na dysku należy zorganizować system plików z jednopoziomowym katalogiem. Elementem katalogu jest opis pliku, zawierajacy co najmniej nazwe, wielkość i sposób rozmieszczenia pliku na wirtualnym dysku. Należy zaimplementować nastepujace operacje, dostepne dla użytkownika programu:

- wyświetlenie zestawienia z aktualna mapa zajetości wirtualnego dysku - czyli listy kolejnych obszarów wirtualnego dysku z opisem: adres, typ obszaru, rozmiar, stan (np. dla bloków danych: wolny/zajety).
- usuwanie wirtualnego dysku
- usuwanie pliku z wirtualnego dysku
- wyświetlanie katalogu dysku wirtualnego
- kopiowanie pliku z dysku wirtualnego na dysk systemu Minix
- kopiowanie pliku z dysku systemu Minix na dysk wirtualny
- tworzenie wirtualnego dysku

Program ma kontrolować wielkość dostepnego miejsca na wirtualnym dysku i pojemność katalogu, reagować na próby przekroczenia tych wielkości.

Nie trzeba realizować funkcji otwierania pliku ani czytania/pisania fragmentów pliku.

Nie trzeba realizować funkcji zwiazanych z współbieżnym dostepem. Zakłada sie dostep sekwencyjny i wyłaczny do wirtualnego dysku.

Należy przygotować demonstracje (zgrupowanie serii poleceń w postaci skryptu interpretera sh) prezentujaca słabe i silne strony przyjetego rozwiazania w kontekście ewentualnych zewnetrznej i wewnetrznej fragmentacji.

Uwaga: Do zdobycia 2 punkty za wejściówke oraz 6 za projekt.

Koncepcja

Jest to bardzo prosty system plików który posiada nastepujace elementy:

1. Super Block
   Przechowuje informacje na temat systemu plików takie jak: rozmiar, ilość bloków, rozmiar przestrzeni użytkownika, jaka cześć przestrzeni użytkownika jest obecnie w użyciu oraz gdzie znajduje sie pierwszy INode.
2. Mapa INodów
   Zawarte sa w niej informacje na temat tego które INode sa obecnie zajete.
3. Mapa Bloków
   Zawarte sa w niej informacje na temat tego które Bloki sa obecnie zajete.
4. Sekcja INode
   Dane które przechowuja kolene INode
5. Sekcja Bloków
   Dane które przechowuja kolene Bloki

Bloki zawieraja dane oraz wskaźniki na kolejny blok przechowujacy dane pliku. Dzieki temu bloki danych pliku nie musza być przechowywane po kolei a co za tym idzie fragmentacja zewnetrzna nie wystepuje. Fragmentacje wewnetrzna da sie zmniejszyć tylko zmniejszajac rozmiar bloków.

Uruchamianie

Plik kompiluje sie jak zwykły plik c

```
cc FileSystem.c
```

Uruchomienie skompilowanego programu wymaga użycia odpowiednich argumentów.

- Tworzenie Systemu plików
  Literka c oznacza create. Argument size oznacza rozmiar systemu plików

  ```
  c size
  ```

- Zapisywanie do systemu plików
  Literki st oznaczaja save to. Argument filename oznacza nazwe pliku który chcemy zapisać w systemie plików

```
st filename
```

- Ekstrachowanie z systemu plików
  Literki sf oznaczaja save from. Argument filename oznacza nazwe pliku który chcemy wyekstrachować z systemu plików

```
sf filename
```

- Usówanie pliku
  Literka r oznacza remove. Argument filename oznacza nazwe pliku który chcemy usunać

```
r filename
```

- Wylistuj wszystkie pliki które sie znajduja w systemie plików
  Literka l oznacza list

```
l
```

- Wypisz informacje na temat stanu pamieci
  Literka m oznacza memory

```
m
```

- Usuń system plików
  Literka d oznacza delete

```
d
```

Przykładowe wywołanie

```
./a.out c 100000
```