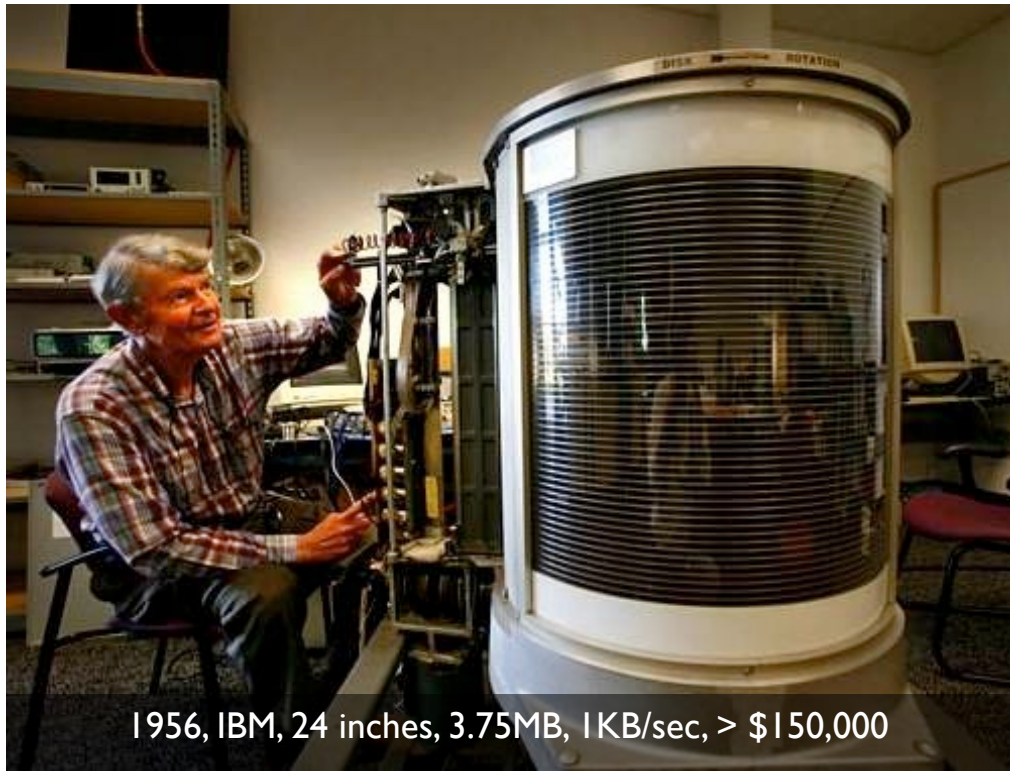
SYSTEMY PLIKÓW

WYKŁAD 4 SYSTEMY OPERACYJNE

AUTOR: DR. INŻ JOANNA KOŁODZIEJCZYK

WYMAGANIA STAWIANE PRZECHOWYWANIU (DŁUGOTERMINOWEMU) INFORMACJI



1956, IBM, 24 inches, 3.75MB, 1KB/sec, > $150,000

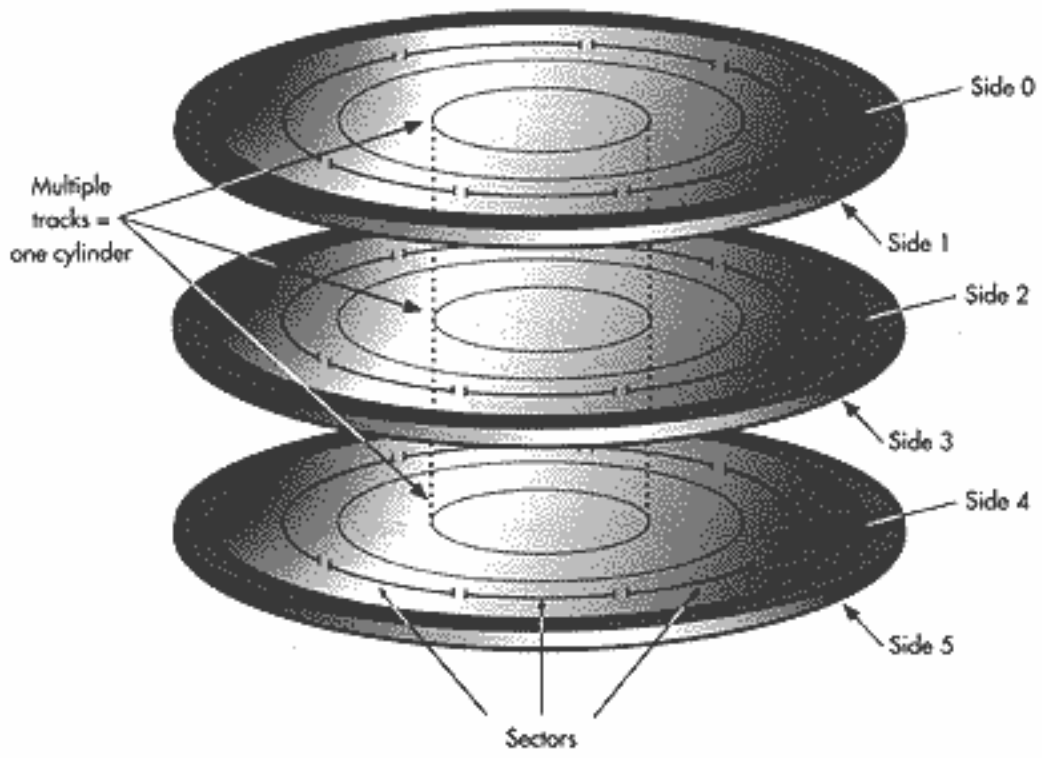
- Musi istnieć możliwość przechowywania bardzo dużej ilości informacji.
- Informacje muszą przetrwać zakończenie procesu z ich wykorzystaniem.
- Wiele procesów musi mieć możliwość jednoczesnego dostępu do informacji.
- Sprzętowo - jedna z najszybciej rozwijających się gałęzi przemysłu
  - Napędzana technologią
  - Napędzana przez popyt:
    - Mainframe storage: IBM, Memorex
    - PC storage: Seagate, DEC, Quantum, etc.
    - Enterprise Storage: EMC, NetApp, etc.
    - Cloud Storage: Dropbox, Google Drive, etc.

HISTORIA TECHNIKI PRZECHOWYWANIA

DYSKI I SYSTEM OPERACYJNY

- Można myśleć o dysku jako o liniowej sekwencji bloków o stałym rozmiarze i wspomagających odczyt i zapis bloków. Zagadnienia:
  - Jak znaleźć informację?
  - Jak utrzymać bezpiecznie dane jednego użytkownika przed innym?
  - Skąd wiadomo który blok jest wolny?
- Zadaniem systemu operacyjnego jest stworzenie abstrakcji pozwalające na:
  - Niskopoziomowe sterowanie urządzeniem (inicjowanie odczytu dysku itp.)
  - Abstrakcje wyższego poziomu (pliki, bazy danych itp.)

DYSK HDD

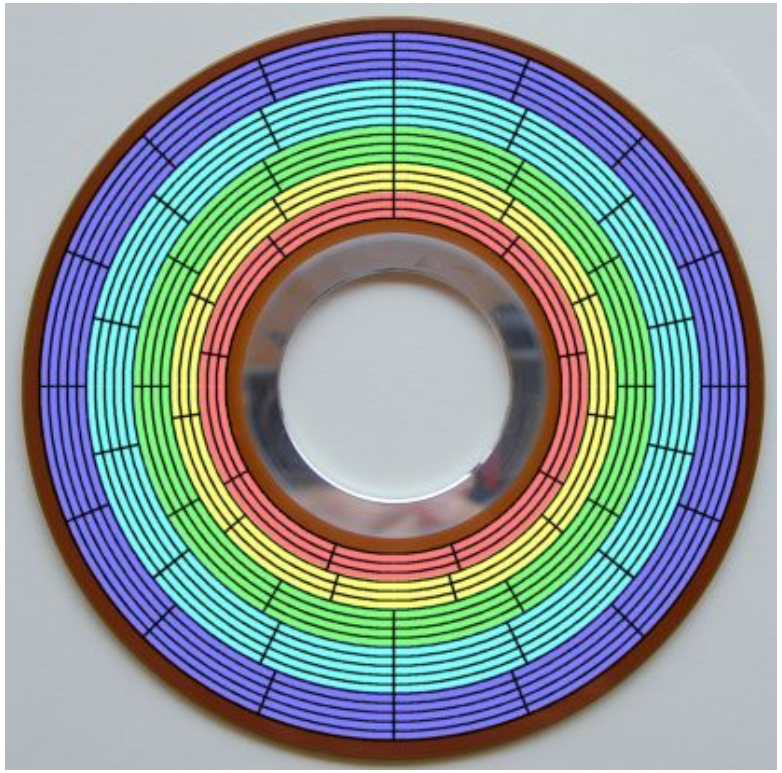


- https://www.youtube.com/watch?v=kdmLvl1n82U
- Elementy dysku twardego
  - Talerze pokryte nośnikiem magnetycznym.
  - Na ramieniu znajdują się głowice zapisująco-odczytujące.
  - Talerze wirują z prędkością 5400 – 10000 obrotów na minutę.
  - Głowice utrzymują się nad powierzchnią talerzy.
  - Informacje zapisywane są w postaci koncentrycznych kręgów zwanych ścieżkami, które z kolei dzielą się na sektory o stałej liczbie bajtów (512bytów). Ten rozmiar jest najmniejszą jednostką dostępu do dysku; w rezultacie co najmniej cały sektor musi być odczytany lub zapisany.
  - Cylinder to ścieżki o tym samym numerze na wszystkich talerzach.

SEKTOR

- Jest najmniejszą porcją informacji, jaka może być czytana lub zapisana na dysku.
- Aby dotrzeć do danego sektora należy określić adresowanie:
  - cylinder, głowicę i sektor (CHS)
  - Logical Block Addressing (LBA) – translacja rzeczywistych numerów głowicy, cylindra i sektora na ich logiczny odpowiednik. – Wystarczy podać #block i czy ma być pisany i czytany
- Przesunięcie głowicy nad właściwą ścieżkę nazywa się czasem przeszukiwania (2-4ms).

ZAPIS WIELOSTREFOWY (MZR)



- **Problem**: sektory znajdujące się dalej od osi dysku będą znacznie dłuższe.
- **Rozwiązanie**: podzielono dysk na kilka stref o określonej liczbie sektorów (od 60 do 120 sektorów na ścieżkę).
- **Zysk**: około 25% większa pojemność i wydajność.
- Strefa niebieska 5 ścieżek z 16 sektorami; strefa cyjan 5 ścieżek z 14 sektorami; zielona strefa: 4 ścieżki z 12 sektorami; żółta: 3 ścieżki z 11 sektorami, czerwona z 3 ścieżkami po 9 sektorów.

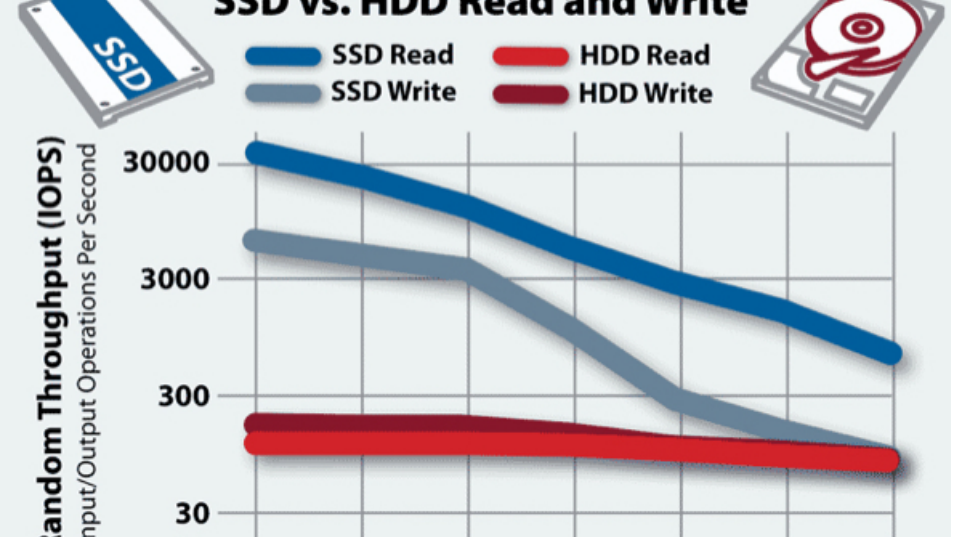
STAŁA PRĘDKOŚĆ KĄTOWA (CAV)

Napęd lub dysk pracujący w trybie CAV utrzymuje stałą prędkość kątową, w przeciwieństwie do prędkości liniowej (CLV).

Typowy napęd CD-ROM działa w trybie CLV, w przeciwieństwie do napędu dyskietek lub dysku twardego, lub gramofonu, który działa w trybie CAV.

W trybie CAV silnik wrzeciona obraca się ze stałą prędkością, co sprawia, że medium przechodzi przez głowicę odczytu/zapisu szybciej, gdy głowica znajduje się na zewnątrz dysku.

Natomiast w trybie CLV prędkość obrotowa wrzeciona zmienia się tak, że nośnik przechodzi przez głowicę z tą samą prędkością, niezależnie od tego, gdzie na dysku znajduje się głowica.



5/16/20

PODSTAWOWE OGRANICZENIE W PRZYPADKU HD

- Mechaniczne
  - Nie można wykonać zbyt szybko odczytu i zapisu.
  - Opóźnienie w milisekundach.

AUTOR: DR. INŻ JOANNA KOŁODZIEJCZYK

https://www.enterprisestorageforum.com/storage-hardware/ssd-vs-hdd.html

SSD vs. HDD

Usually 10,000 or 15,000 rpm SAS drives

SSD		HDD
0.1 ms	**Access Times** SSDs exhibit virtually no access time	5.5-8.0 ms
SSDs deliver at least 6000 io/s	**Random I/O Performance** SSDs are at least 15 times faster than HDDs	HDDs reach up to 400 io/s
SSDs have a failure rate of less than 0.5%	**Reliability** This makes SSDs 4-10 times more reliable	HDDs failure rate fluctuates between 2-5%
SSDs consume between 2 and 5 watts	**Energy Savings** This means that on a large server, approximately 100 watts are saved	HDDs consume between 6 and 15 watts
SSDs have an average I/O wait of 1%	**CPU Power** You will have an extra 6% of CPU power for other operations	HDDs average I/O wait is about 7%
The average service time for an I/O request while running a backup remain below 20 ms	**Input/Output Request Times** SSDs allow for much faster data access	The I/O request time with HDDs during backup rises up to 400-500 ms
SSD backups take about 6 hours	**Backup Rates** SSDs allow for 3-5 times faster	HDD backups take up to 20-24 hours

DYSK SSD

- W przeciwieństwie do dysku twardego, dysk SSD składa się z półprzewodnikowych bloków pamięci, nie zawiera części mechanicznych.
- Najmniejszą jednostką dysku SSD jest strona, która składa się z kilku komórek pamięci i ma zazwyczaj rozmiar 4 KB.
- Kilka stron na dysku SSD jest podsumowanych w jednym bloku.
- Blok jest najmniejszą jednostką dostępu w dysku SSD.
- Obecnie 128 stron jest w większości łączonych w jednym bloku; dlatego blok zawiera 512 KB.

Wydajność zapisu losowego

Aby zapisać, należy najpierw usunąć duże strony (32-64 strony) i przeprogramować.

Wypalenie

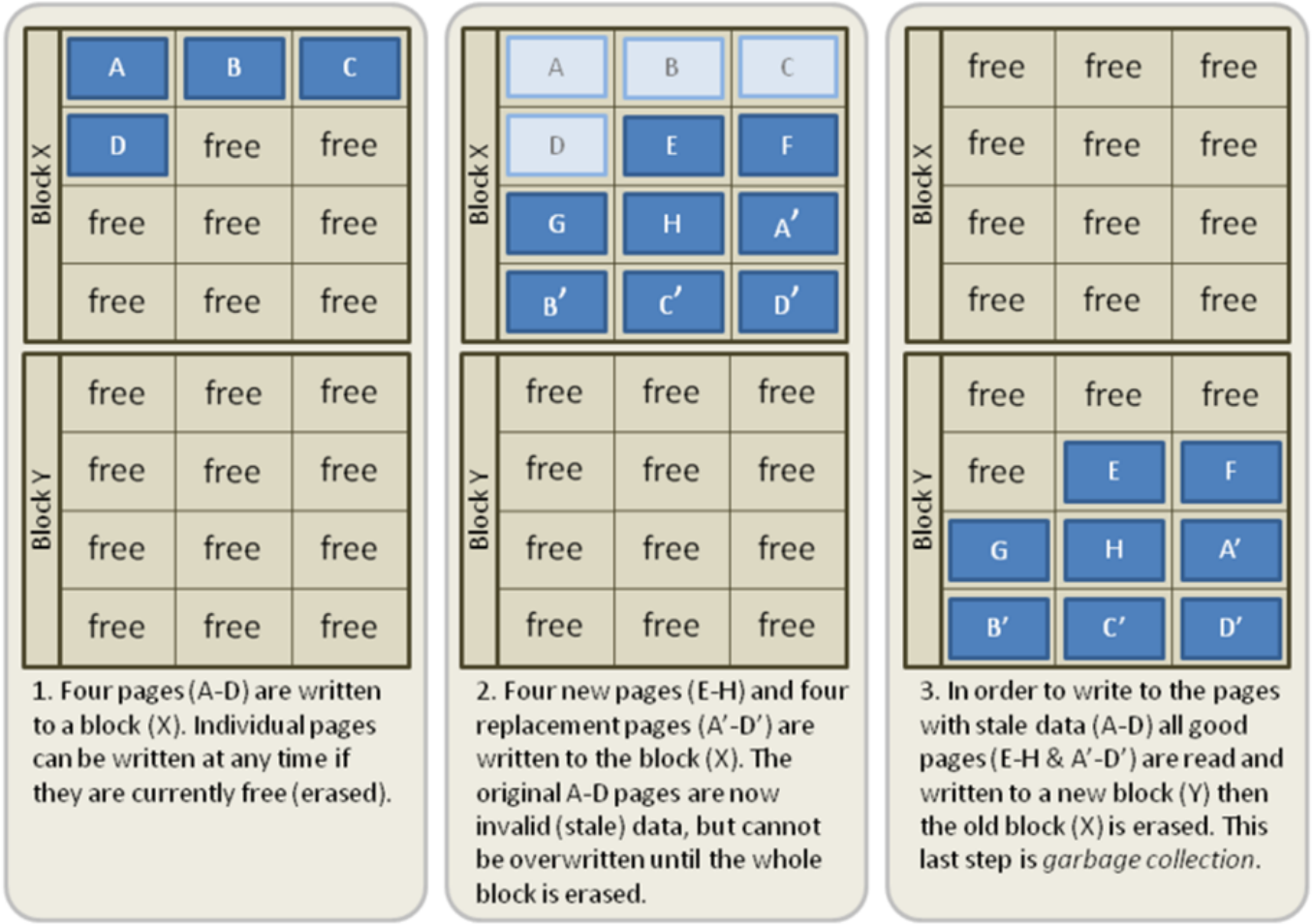
Każda komórka ma ograniczone cykle wymazywania/programowania.

- Zakres od 1.000 - 100.000 wpisów

OGRANICZENIA SSD

ZAPIS I ODCZYT SSD

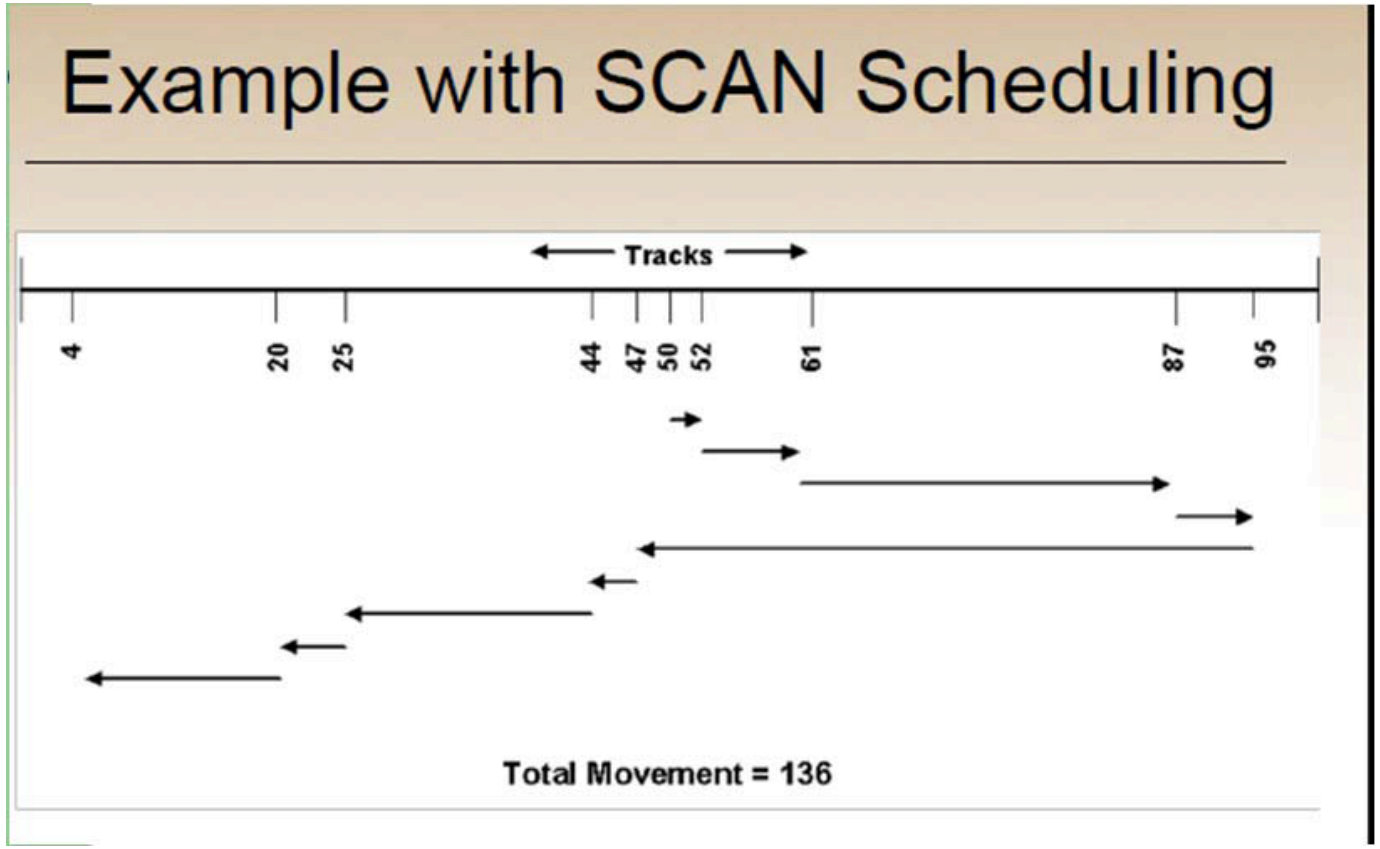
- W przeciwieństwie do dysków twardych, używają innego algorytmu do określenia pierwszego sektora logicznego. Dane są zawsze odczytywane i zapisywane w blokach. Dlatego partycje/wolumeny muszą być zawsze widziane w blokach.



PLANOWANIE ŻĄDAŃ DO DYSKU

- System operacyjny próbuje zaplanować żądania, które są ustawiane w kolejce oczekującej na dysk.
  - FCFS - Rozsądne, gdy obciążenie jest niskie. - Długi czas oczekiwania na długie kolejki zapytań.
  - SSTF (pierwszy jest ten, a najkrótszym czas wyszukiwania) - Minimalizuje poszukiwanie, maksymalizuj szybkość żądania. W HDD aworyzuje bloki środkowe.
  - SCAN (winda) - Zlecenia serwisowe w jednym kierunku aż do wykonania, a następnie odwrotnie.
  - C-SCAN - Jak SCAN, ale tylko w jednym kierunku (maszyna do pisania)

SCAN HDD


Example with SCAN Scheduling
Tracks
4
20
25
44
47
50
52
61
87
95
Total Movement = 136

PLANOWANIE KOLEJEK ŻĄDAŃ

- O ile nie ma dużych kolejek żądań, algorytm planowania dysku nie ma większego wpływu na wydajność.
- Ważne dla serwerów, mniej dla komputerów PC.
- Nowoczesne dyski często same planują (wspomaganie sprzętowe z algorytmem) - dyski znają swój układ lepiej niż system operacyjny, potrafią lepiej optymalizować.
- Wtedy ignoruje się, cofa każde planowanie wykonywane przez system operacyjny.

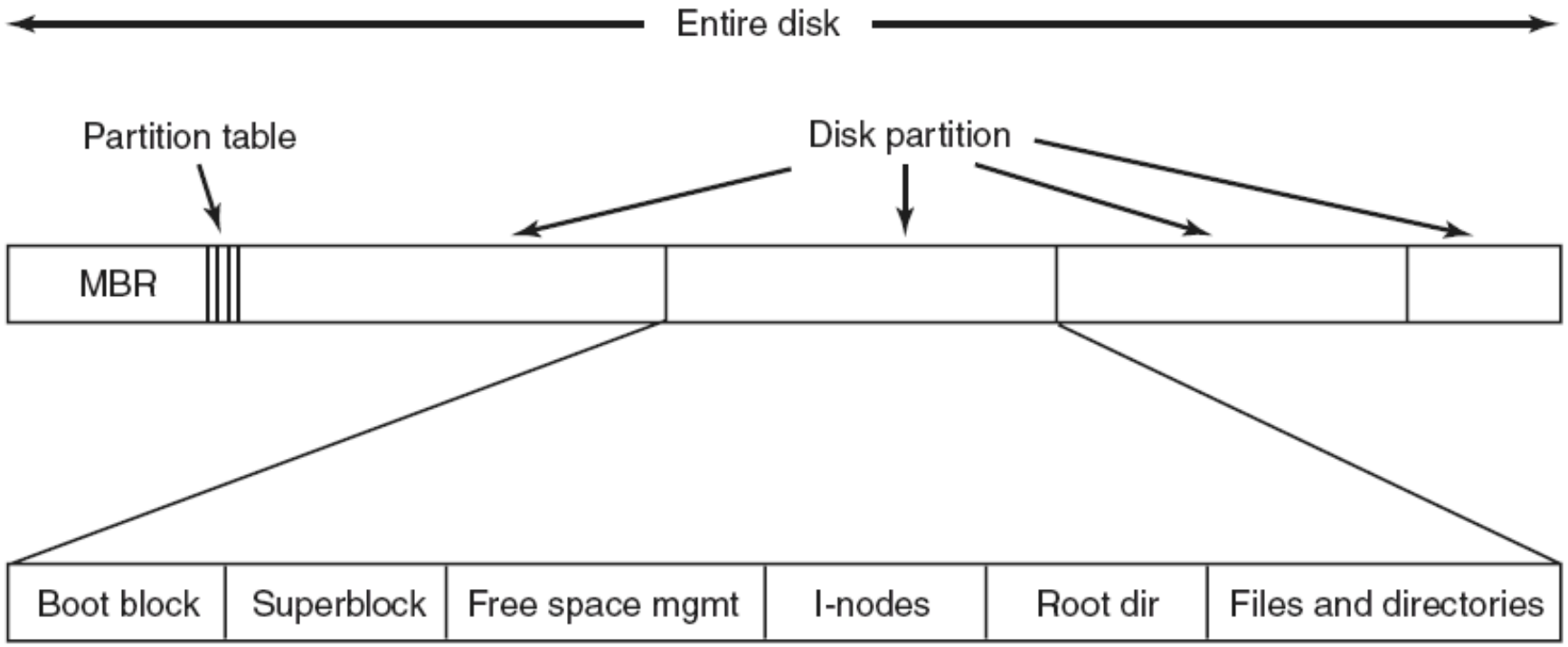
LOGICZNA STRUKTURA DYSKU

- System plików to logiczna struktura na dysku oraz procedury stosowane, by kontrolować dostęp do danych na dysku.
- Różne systemy operacyjne stosują różne sposoby organizowania i kontrolowania dostępu do danych zapisanych na dysku.
- System plików jest niezależny od hardware.
- Struktura logiczna ma wpływa na:
  - osiągi,
  - niezawodność,
  - rozszerzalność systemu magazynowania danych.

SYSTEM PLIKÓW

- Sposób, w jaki komputer organizuje pliki i katalogi na nośniku danych o swobodnym dostępie.
- Określa, jak informacje są zapisywane i odczytywane.
- Definiuje także wielkość klastrów, możliwe do użycia atrybuty plików i system poprawnych nazw plików i katalogów.
- Popularne systemy plików:
  - DOS i Windows
    - FAT16
    - FAT32
    - NTFS
  - Linux
    - Ext3
    - Ext4
    - ReiserFS

MOŻLIWE UKŁADY SYSTEMU PLIKÓW

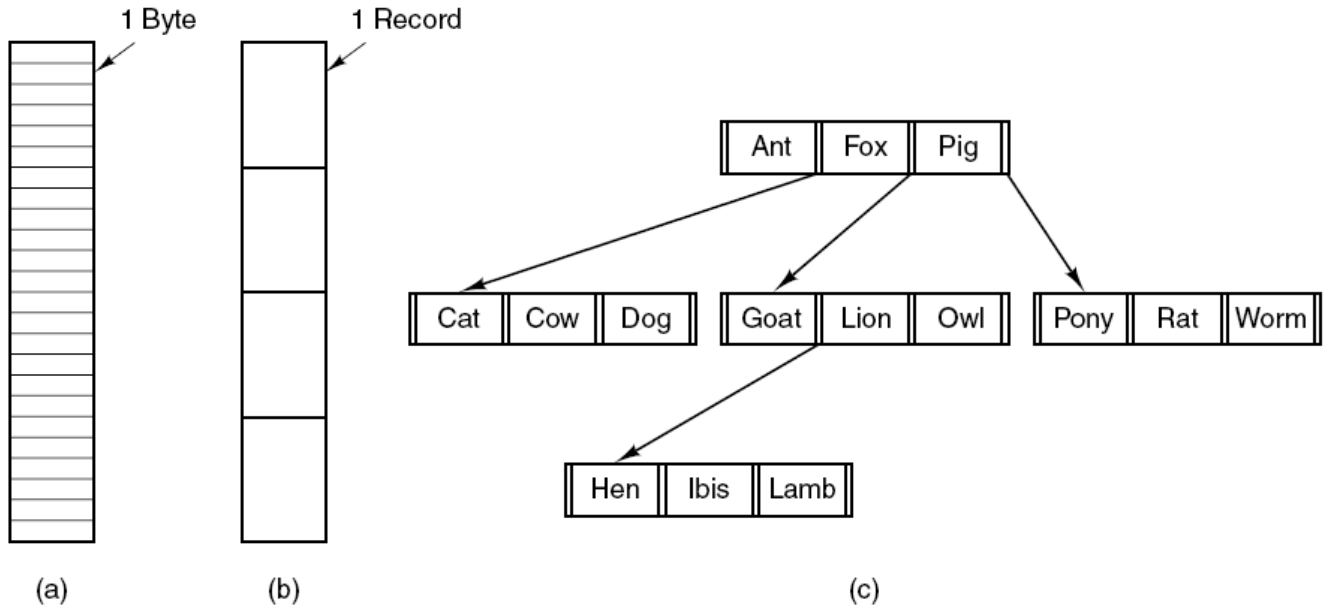


PLIK

- Zbiór powiązanych ze sobą informacji określonych przez twórcę. Najczęściej pliki reprezentują programy lub dane do programów.
- Plik może mieć również typ
  - Rozumiane przez inne części systemu operacyjnego lub biblioteki uruchomieniowe - Wykonywalne, dll, souce, obiekt, tekst itp.
  - Zrozumiałe dla systemu plików - Blok/znak urządzenie, katalog, link, itp.
- Plik jest ciągiem bitów, bajtów, wierszy, rekordów, których znaczenie określa twórca.
- Plik ma określoną strukturę zależną od typu pliku (plik tekstowy, źródłowy, wykonywalny).

TYPY PLIKÓW – STRUKTURA

- Trzy typy plików:
  - (a) sekwencja bajtów
  - (b) sekwencja rekordów
  - (c) drzewo



NAZWY PLIKÓW

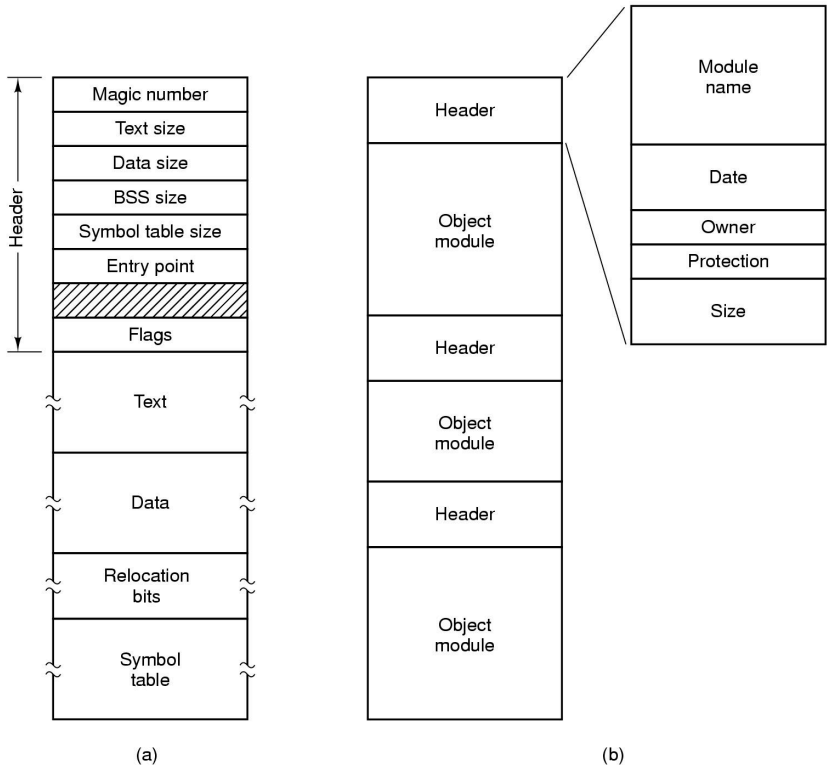
Extension	Meaning
file.bak	Backup file
file.c	C source program
file.gif	Compuserve Graphical Interchange Format image
file.hlp	Help file
file.html	World Wide Web HyperText Markup Language document
file.jpg	Still picture encoded with the JPEG standard
file.mp3	Music encoded in MPEG layer 3 audio format
file.mpg	Movie encoded with the MPEG standard
file.o	Object file (compiler output, not yet linked)
file.pdf	Portable Document Format file
file.ps	PostScript file
file.tex	Input for the TEX formatting program
file.txt	General text file
file.zip	Compressed archive

ATRYBUTU PLIKU

- nazwa (system operacyjny określa długość nazwy i stosowane znaki),
- typ (system operacyjny może rozpoznawać typy plików (pliki muszą być zawsze z zakresu plików zarejestrowanych) lub nie),
- czas założenia,
- nazwa twórcy lub inny jego identyfikator,
- długość i in.

Attribute	Meaning
Protection	Who can access the file and in what way
Password	Password needed to access the file
Creator	ID of the person who created the file
Owner	Current owner
Read-only flag	0 for read/write; 1 for read only
Hidden flag	0 for normal; 1 for do not display in listings
System flag	0 for normal files; 1 for system file
Archive flag	0 for has been backed up; 1 for needs to be backed up
ASCII/binary flag	0 for ASCII file; 1 for binary file
Random access flag	0 for sequential access only; 1 for random access
Temporary flag	0 for normal; 1 for delete file on process exit
Lock flags	0 for unlocked; nonzero for locked
Record length	Number of bytes in a record
Key position	Offset of the key within each record
Key length	Number of bytes in the key field
Creation time	Date and time the file was created
Time of last access	Date and time the file was last accessed
Time of last change	Date and time the file was last changed
Current size	Number of bytes in the file
Maximum size	Number of bytes the file may grow to

TYPY PLIKÓW



- a – wykonywalny z wczesnego UNIXa.
  - Jest po prostu sekwencją bajtów
  - SO wykona go, jeżeli będzie miał prawidłowy format.
  - Plik składa się z pięciu sekcji: nagłówka tekstu danych bitów relokacji i tabeli symboli.
  - Magiczna liczba identyfikuje plik jako wykonywalny.
- b – archiwum
  - Przykład pliku binarnego.
  - Składa się on z kolekcji procedur bibliotecznych które są skompilowane ale nie są połączone.
  - Nagłówki modułów pełne są liczb binarnych.

PODSTAWOWE OPERACJE (WYWOŁANIA SO) ZWIĄZANE Z PLIKAMI

UNIX	WINDOWS
create(name)	CreateFile(name, CREATE)
open(name, how)	CreateFile(name, OPEN)
read(fd, buf, len)	ReadFile(handle, …)
write(fd, buf, len)	WriteFile(handle, …)
sync(fd)	FlushFileBuffers(handle, …)
seek(fd, pos)	SetFilePointer(handle, …)
close(fd)	CloseHandle(handle, …)
unlink(name)	DeleteFile(name)
Delete(name)	CopyFile(name)
	MoveFile(name)

PROGRAM DO KOPIOWANIA PLIKU (WYKORZYSTANIEM FUNKCJI SO)

```
/* File copy program. Error checking and reporting is minimal. */

#include <sys/types.h>                  /* include necessary header files */
#include <fcntl.h>
#include <stdlib.h>
#include <unistd.h>

int main(int argc, char *argv[]);       /* ANSI prototype */

#define BUF_SIZE 4096                   /* use a buffer size of 4096 bytes */
#define OUTPUT_MODE 0700                /* protection bits for output file */

int main(int argc, char *argv[])
{
    int in_fd, out_fd, rd_count, wt_count;
    char buffer[BUF_SIZE];

    if (argc != 3) exit(1);             /* syntax error if argc is not 3 */

    /* Open the input file and create the output file */
    in_fd = open(argv[1], O_RDONLY);    /* open the source file */
    if (in_fd < 0) exit(2);             /* if it cannot be opened, exit */
    out_fd = creat(argv[2], OUTPUT_MODE); /* create the destination file */
    if (out_fd < 0) exit(3);            /* if it cannot be created, exit */

    /* Copy loop */
    while (TRUE) {
        rd_count = read(in_fd, buffer, BUF_SIZE); /* read a block of data */
        if (rd_count <= 0) break;       /* if end of file or error, exit loop */
        wt_count = write(out_fd, buffer, rd_count); /* write data */
        if (wt_count <= 0) exit(4);     /* wt_count <= 0 is an error */
    }

    /* Close the files */
    close(in_fd);
    close(out_fd);
    if (rd_count == 0)                  /* no error on last read */
        exit(0);
    else
        exit(5);                        /* error on last read */
}
```

KLASTER

- Urządzenie podzielony jest na niewielkie fragmenty, zwane **jednostkami alokacji** lub **klastrami**.
- Klaster jest wartością logiczną, a nie fizyczną (nie jest to sektor, czy blok).
- Klaster jest najmniejszą porcją zapisywaną lub czytaną przez OS na urządzeniu pamięci zewnętrznej.
- Rozmiar klastra można zdefiniować w zakresie od 512B do 64 kB.
- Każdy znajdujący się na urządzeniu plik zajmuje jeden bądź więcej klastrów.
- W jednym klastrze może znajdować się tylko jeden plik (lub jego część).

FRAGMENTACJA WEWNĘTRZNA

Gdy rozmiar pliku jest mniejszy niż rozmiar klastra, niewykorzystane miejsce marnuje się.

np.. klaster 512B plik ma rozmiar1949B

(4 klastry * 512 = 2048B)
=>

(2048B-1949B = 99B) 99 B zmarnowane.

Im mniejsze jednostki alokacji, tym bardziej ekonomicznie wykorzystane jest miejsce na rządzeniu. Maksymalna liczba jednostek alokacji (przypadająca na plik) jest jednak ograniczona przez system plików.

ZBYT MAŁE JEDNOSTKI ALOKACJI

Gdyby dysk miał 2GB, a jednostki alokacji równe były 512B to składałby się z 4 milionów pojedynczych sektorów/bloków.

Śledzenie połączeń tak małych fragmentów jest nieefektywne.

WIELKOŚĆ KLASTRÓW W RÓŻNYCH SYSTEMACH PLIKÓW

Rozmiar partycji		FAT16	FAT32	NTFS
0-32	MB	0,5kB	-	0,5kB
33-64	MB	1kB	-	0,5kB
65-127	MB	2kB	-	0,5kB
128-255	MB	4kB	-	0,5kB
256-511	MB	8kB	-	0,5kB
512-1023	MB	16kB	4kB	1kB
1--2	GB	32kB	4kB	2kB
2--4	GB	64kB	4kB	4kB
4--8	GB	-	4kB	8kB
8--16	GB	-	8kB	16kB
16-32	GB	-	16kB	32kB
>32	GB	-	32kB	64kB

METODY PRZYDZIAŁU MIEJSCA NA DYSKU

DESKRYPTOR PLIKU

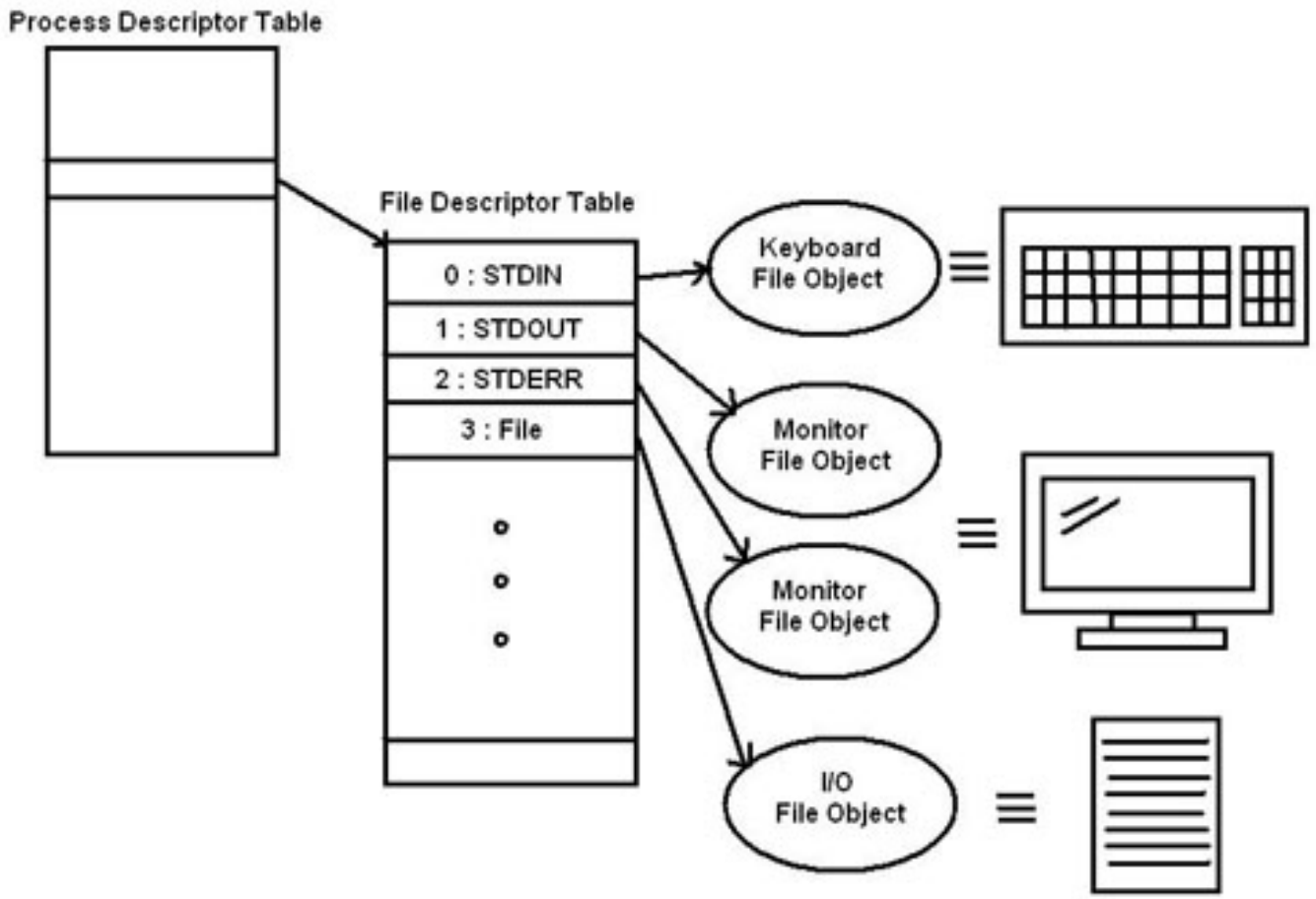
- **Deskryptor pliku** – identyfikator pliku wykorzystywany przez system operacyjny. Po wykonaniu operacji otwarcia pliku, deskryptor pliku może być wykorzystywany wielokrotnie przez wywołanie systemowe w operacjach wejścia/wyjścia. Deskryptor pliku jest zwracany przez funkcje systemowe z rodziny *open*, w odróżnieniu od funkcji *fopen*, które zwracają uchwyt pliku. Uchwyt pliku jest strukturą danych zawierającą dodatkowe informacje na temat otwartego pliku.
- Informacje o deskryptorach przechowywane są na dysku.
- Aplikacja nie może bezpośrednio czytać z i pisać do tablicy deskryptorów plików, zmiany w niej są dokonywane przez system w zależności od wykonywanej operacji na pliku.
- Deskryptor opisująca, które klastry na urządzeniu reprezentują dany plik

DESKRYPTOR PLIKU

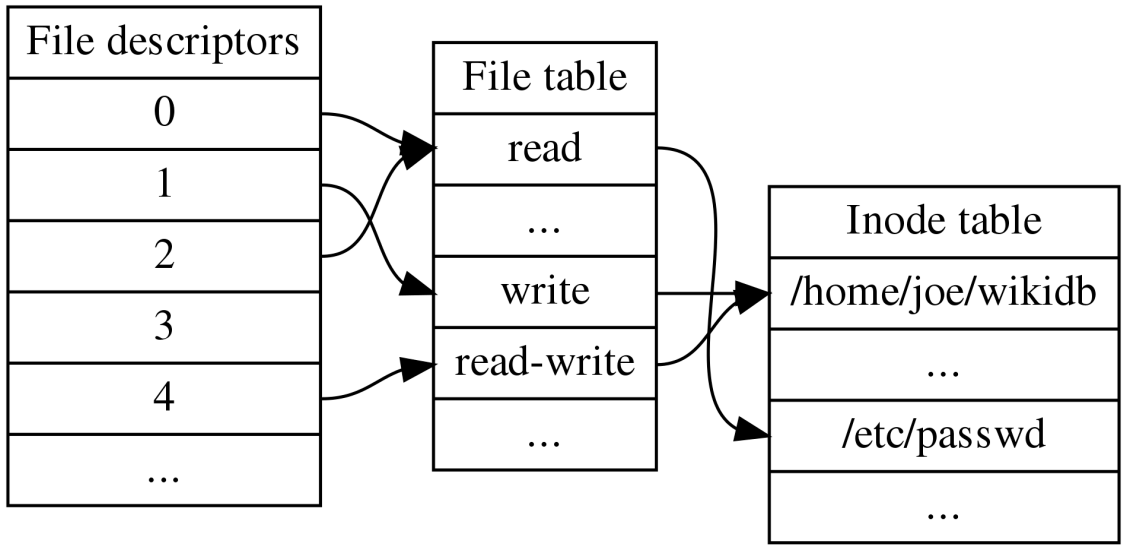
Wartość deskryptora	Nazwa	Nazwa uchwytu
0	Standardowe wejście (stdin)	STDIN
1	Standardowe wyjście (stdout)	STDOUT
2	Standardowe wyjście diagnostyczne (stderr)	STDERR

- Określenie deskryptor pliku jest używane głównie w systemach operacyjnych zgodnych z normą POSIX. W terminologii Microsoft Windows używane jest określenie "uchwyt pliku" (ang. file handle).
- Zgodnie z POSIX deskryptor pliku to liczba całkowita, czyli wartość typu int z języka C. Domyślnie każdy proces po uruchomieniu ma otwarte 3 standardowe deskryptory plików:

DESKRYPTOR PLIKU



DESKRYPTOR PLIKU DLA POJEDYNCZEGO PROCESU TABELI PLIKÓW I TABELI I-NODE



- Wiele deskryptorów plików może odnosić się do tej samej pozycji tabeli plików (np. w wyniku podwójnego wywołania systemowego)
- Wiele pozycji tabeli plików może z kolei odnosić się do tego samego i-node (jeśli był wielokrotnie otwierany; tabela jest nadal uproszczona, ponieważ reprezentuje i-nody przez nazwy plików, nawet jeśli i-node może mieć wiele nazw). Deskryptor pliku 3 nie odnosi się do niczego w tabeli plików, co oznacza, że został zamknięty.

Plik zajmuje kolejne bloki na dysku.

Adresy definiują uporządkowanie liniowe.

Dostęp do bloku $b+1$ po bloku b jest najszybszy.

W przypadku HDD ruch głowicy nastąpi przy zmianie ścieżki (jest to jednak kolejna sąsiednia ścieżka).

ALGORYTM PRZYDZIAŁU MIEJSCA NA DYSKU CIĄGŁY

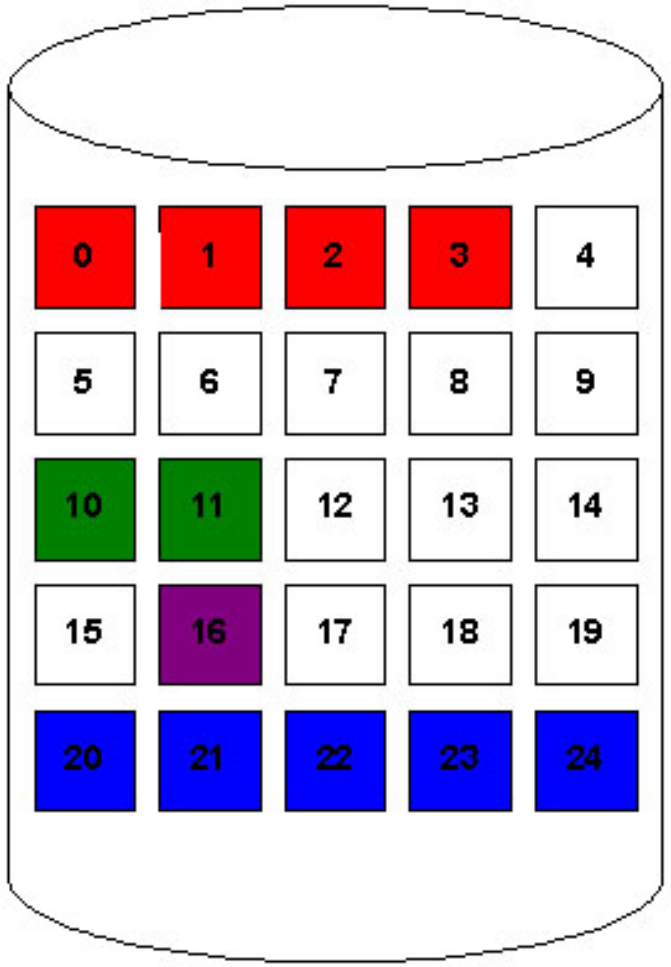
PRZYDZIAŁ CIĄGŁY

W deskryptorze pliku pamięta się adres początkowy pliku i ilość zajmowanych bloków.

Adres początkowy: *b*
liczba bloków: *n*
zajmowane kolejne bloki:

b, b+1, b+2, ..., b+n-1

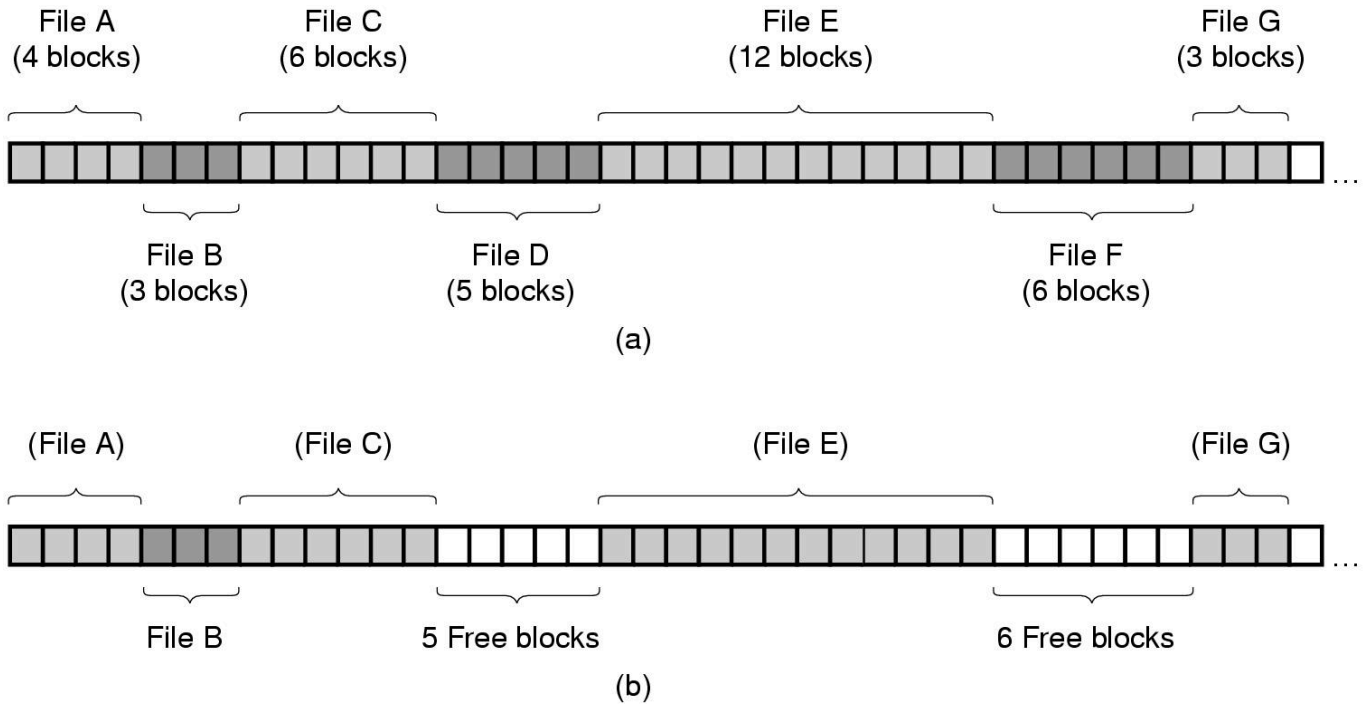
PRZYDZIAŁ CIĄGŁY



Plik	Początek	Długość
pierwszy	0	4
drugi	10	2
trzeci	16	1
czwarty	20	5

PRZYDZIAŁ CIĄGŁY

- Schemat a) ciągła alokacja miejsca na dysku dla 7 plików
- Schemat b) Pliki D i F zostały usunięte



DOSTĘP DO PLIKU W PRZYDZIALE CIĄGŁYM

dostęp sekwencyjny: do adresu bloku ostatnio czytanego wystarczy dodać 1.

dostęp swobodny (random): by dotrzeć do *k*-tego bloku wystarczy dodać do adresu początkowego *k*.

WADY PRZYDZIAŁY CIĄGŁEGO

- Znalezienie miejsca na dysku na zapisanie całego pliku: należy znaleźć *n*-wolnych leżących obok siebie bloków.
- Powstaje fragmentacją zewnętrzną.
- Tworzone pliki muszą mieć określony z góry rozmiar, by określić wielkość przydzielanej dla niego przestrzeni dyskowej.
- Rozwiązaniem może być przepisywanie pliku do większych wolnych obszarów, co spowalnia pracę systemu operacyjnego.

METODY WYSZUKIWANIA WOLNYCH OBSZARÓW W PRZYDZIALE CIĄGŁYM

- pierwszy pasujący
- najgorszy pasujący (największy pusty obszar jest przydzielany)
- najlepszy pasujący (obszar o liczbie pustych bloków najlepiej liczebnie dopasowany do pliku)

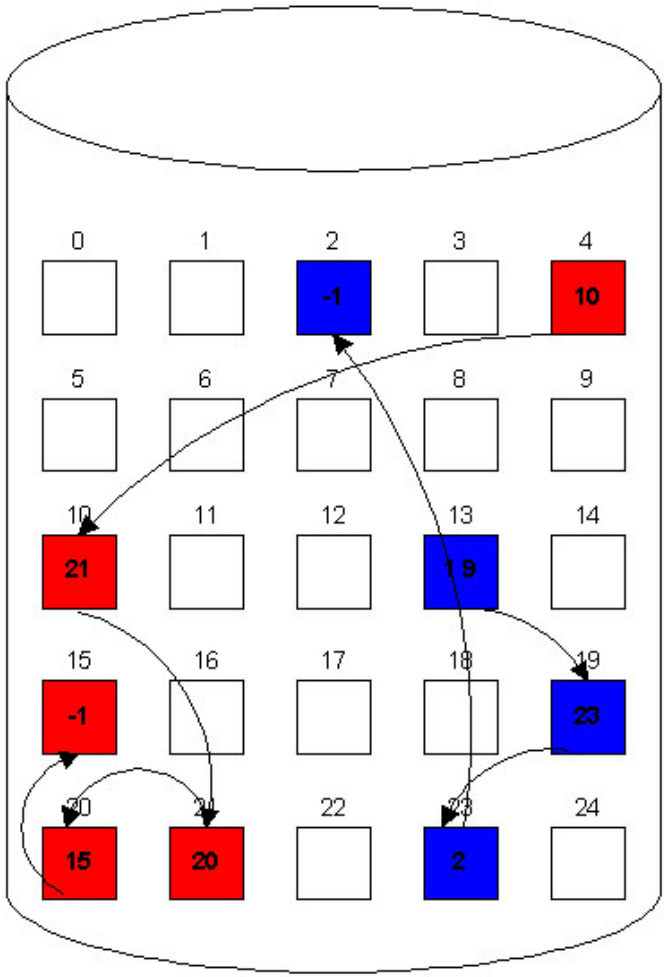
PRZYDZIAŁ LISTOWY – LISTA JEDNOKIERUNKOWA

Każdy plik stanowi listę powiązanych bloków, które mogą znajdować się gdziekolwiek na dysku.

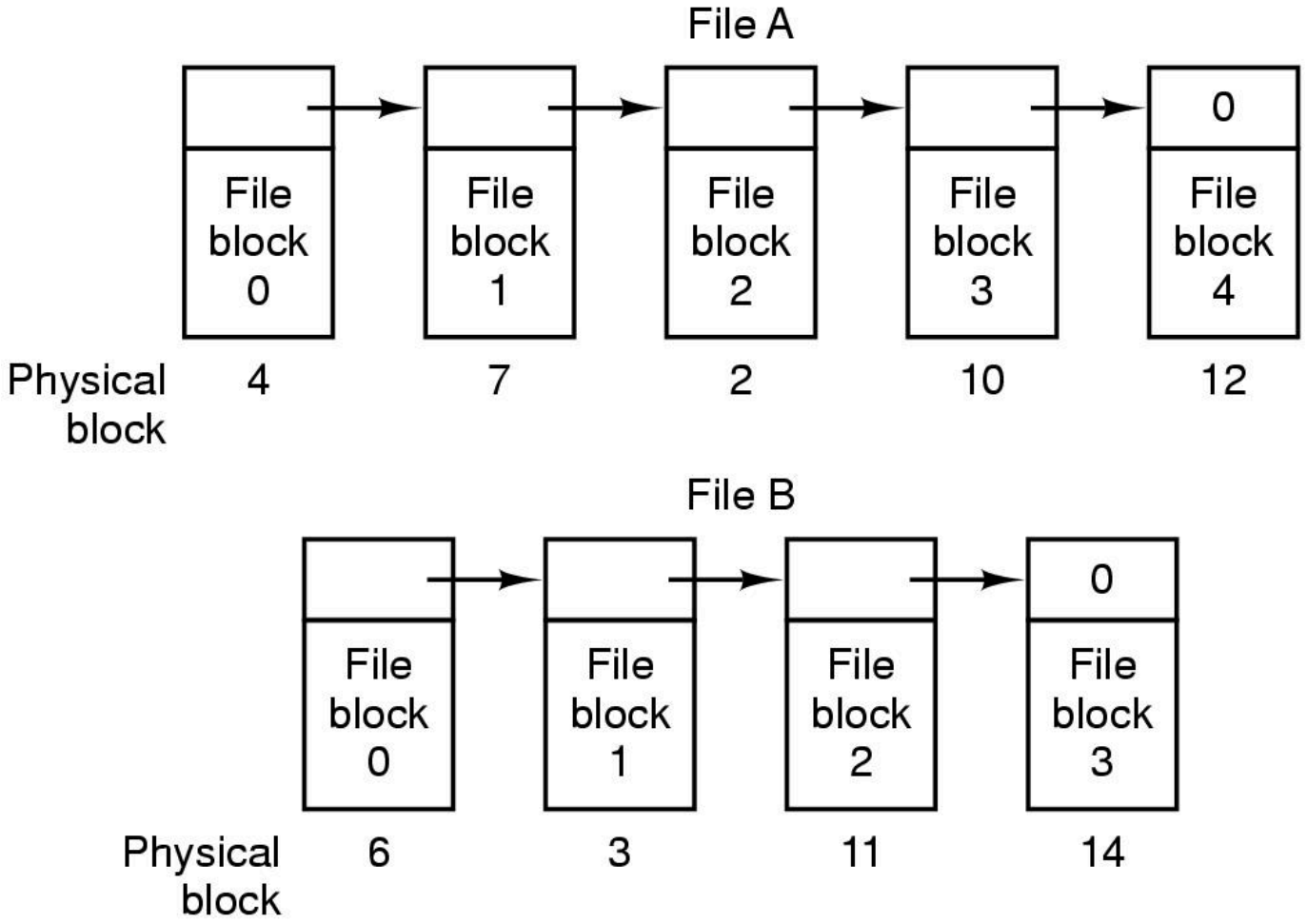
W deskryptorze pliku zapisane są wskaźniki do pierwszego i ostatniego bloku pliku.

Każdy blok zawiera wskaźnik do następnego bloku (część bajtów z bloku jest przeznaczona na pamiętanie wskaźnika).

PRZYDZIAŁ LISTOWY



Plik	Początek	Koniec
plik1	13	2
plik2	4	15

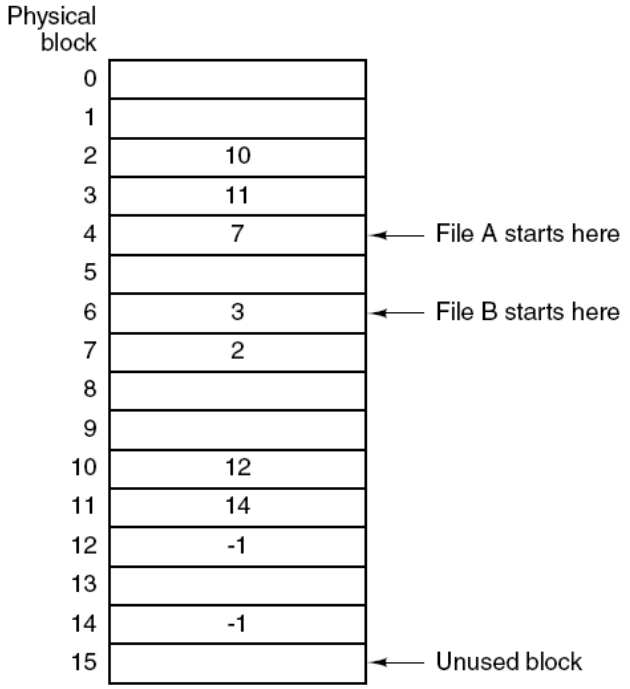


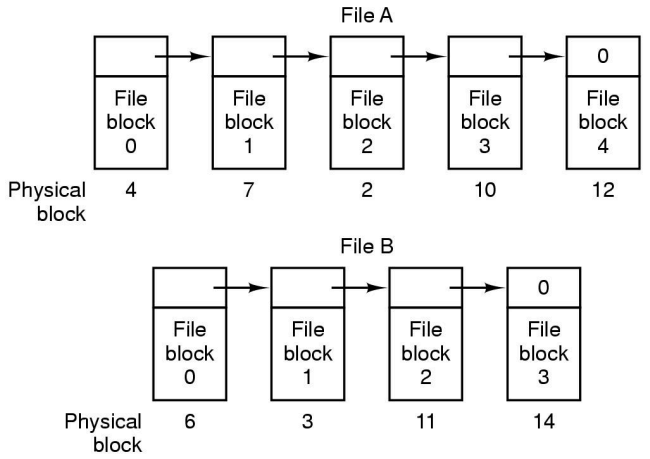
PRZYDZIAŁ LISTOWY

ZAPISYWANIE PLIKU W POSTACI JEDNOKIERUNKOWEJ LISY BLOKÓW DYSKOWYCH.

ALOKACJA Z LISTĄ Z WYKORZYSTANIEM TABELI

- Plik A wykorzystuje bloki dyskowe 4,7,2,10,12.





TWORZENIE PLIKU W PRZYDZIALE LISTOWYM

- Utworzenie nowego deskryptora pliku (zapamiętaj adres pierwszego bloku plików).
- Zawartość bloku danych jest pusta.
- Pisanie powoduje pobranie pierwszego wolnego bloku i usunięcie go z listy wolnych bloków i zapisanie w nim informacji.
- Nowy blok zostanie dowiązany do końca pliku.

ZALETY I WADY PRZYDZIAŁU LISTOWEGO

Zalety

- Brak zewnętrznej fragmentacji.
- Brak konieczności określania wielkości pliku.
- Nie ma potrzeby upakowywania dysku.

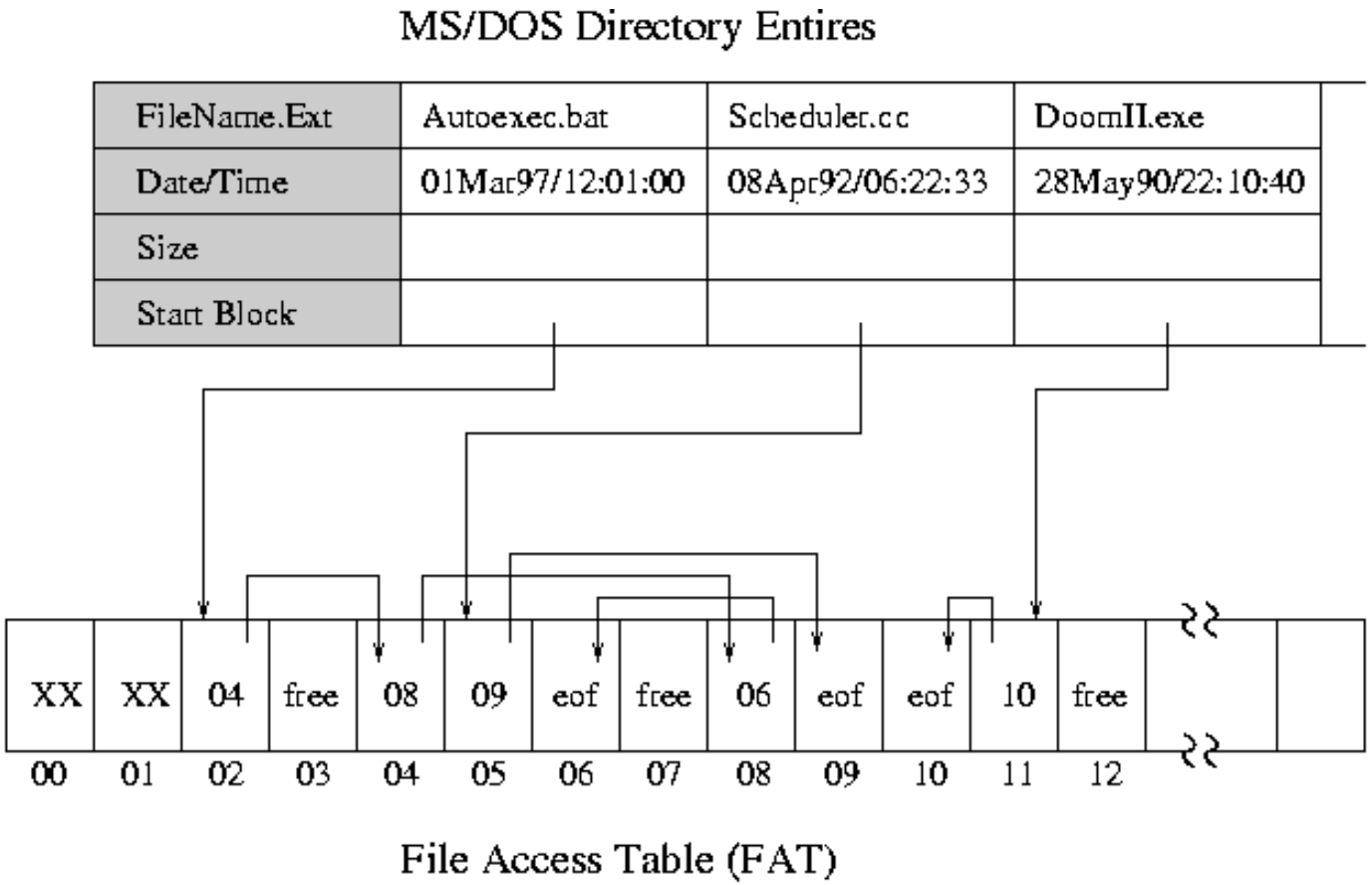
Wady

- Zastosowanie tylko do plików o dostępie sekwencyjnym,
- Przestrzeń zajmowana przez wskaźniki: pliki poprzez zapamiętywanie wskaźników na kolejne bloki rosną,
- Zniszczenie lub zgubienie jednego ze wskaźników (przez błąd systemu operacyjnego lub urządzenia) powoduje stratę pliku. [Stosuje się duplikowanie list powiązań lub pamiętanie w każdym bloku nazwy pliku i względnego numeru bloku].

TABLICA PRZYDZIAŁU PLIKÓW (FAT)

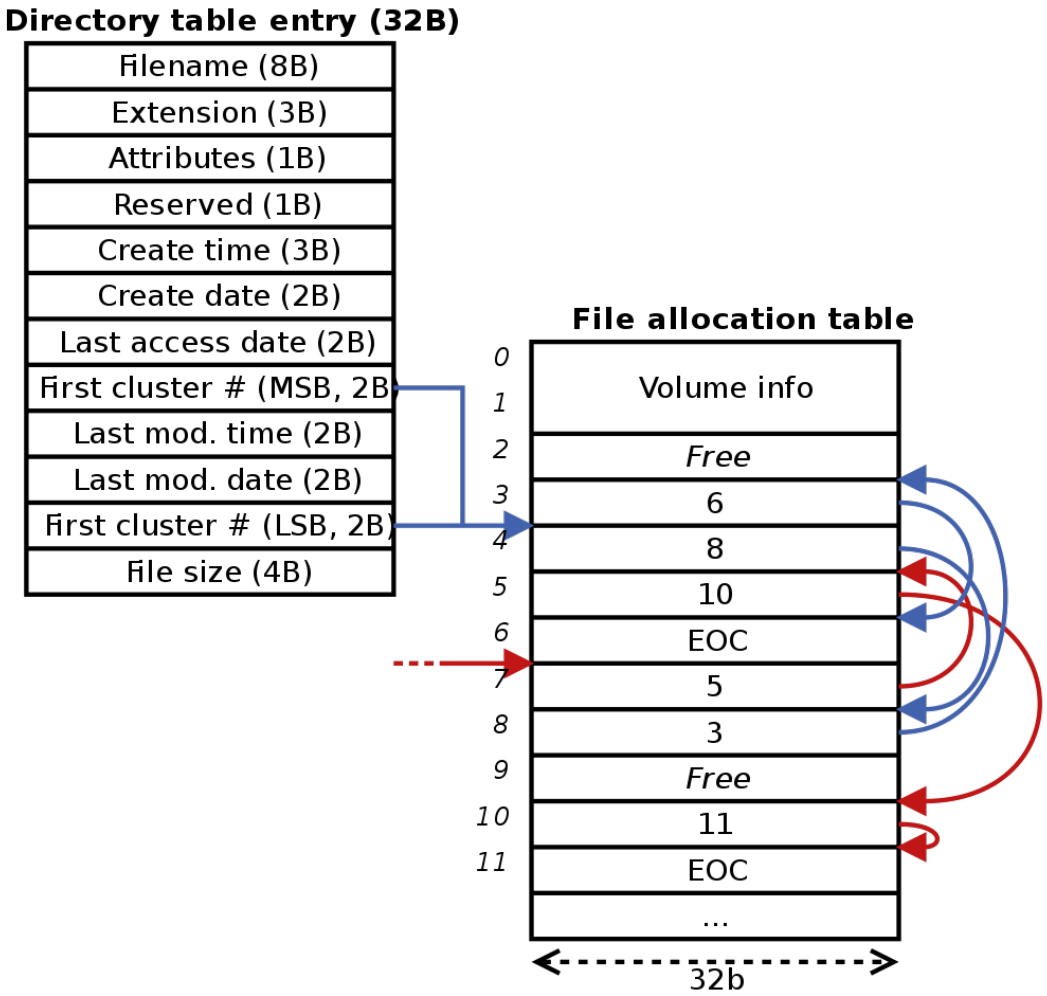
- Jest to odmiana przydziału listowego.
- Na dysku wydziela się sekcję zawierającą tablicę FAT.
- Pozycja w tablicy odpowiada blokowi i jest indeksowana numerami bloków.
- Deskryptor zawiera numer pierwszego bloku pliku. Pozycja w tablicy indeksowana numerem tego bloku zawiera numer bloku następnego.
- W ostatnim bloku znajduje się specjalny znacznik końca pliku.
- Bloki nieużywane mają numer wskaźnika równy zero.

FAT



http://www.cs.wisc.edu/~bart/537/lecturenotes/s23.html

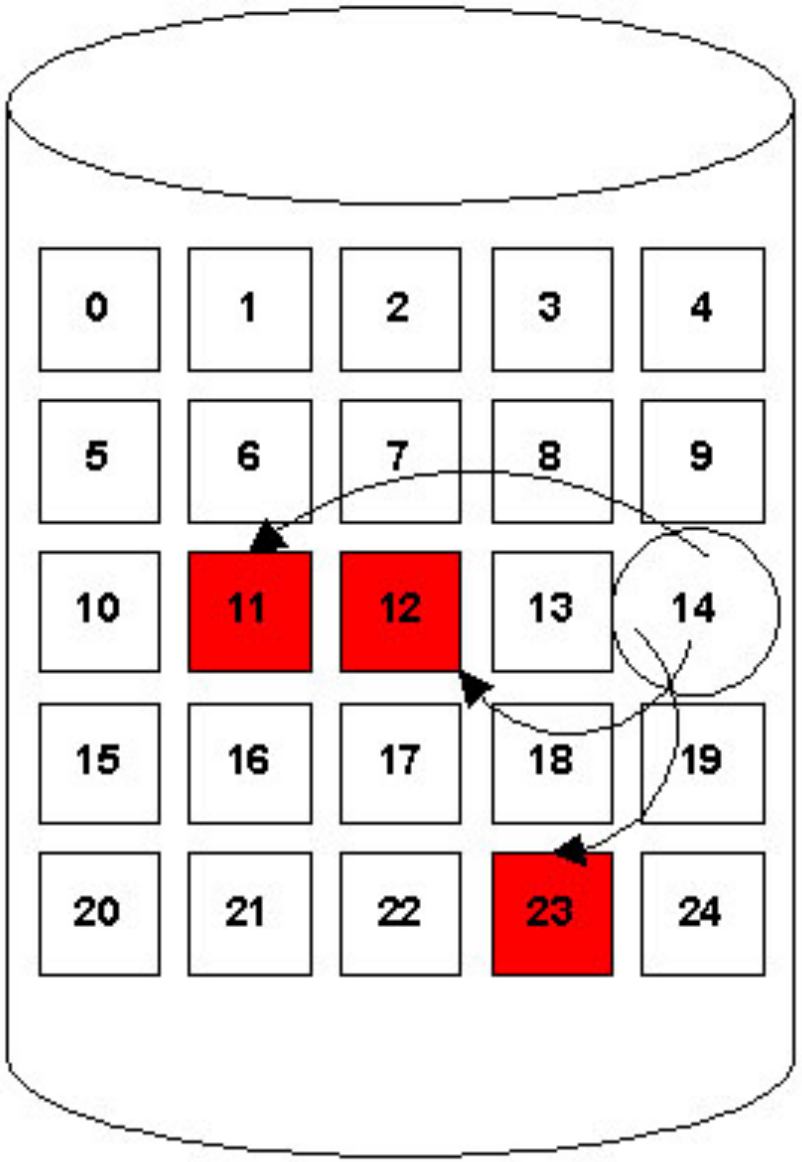
FILE ALLOCATION TABLE AND DIRECTORY TABLE FOR A FAT32 FILESYSTEM

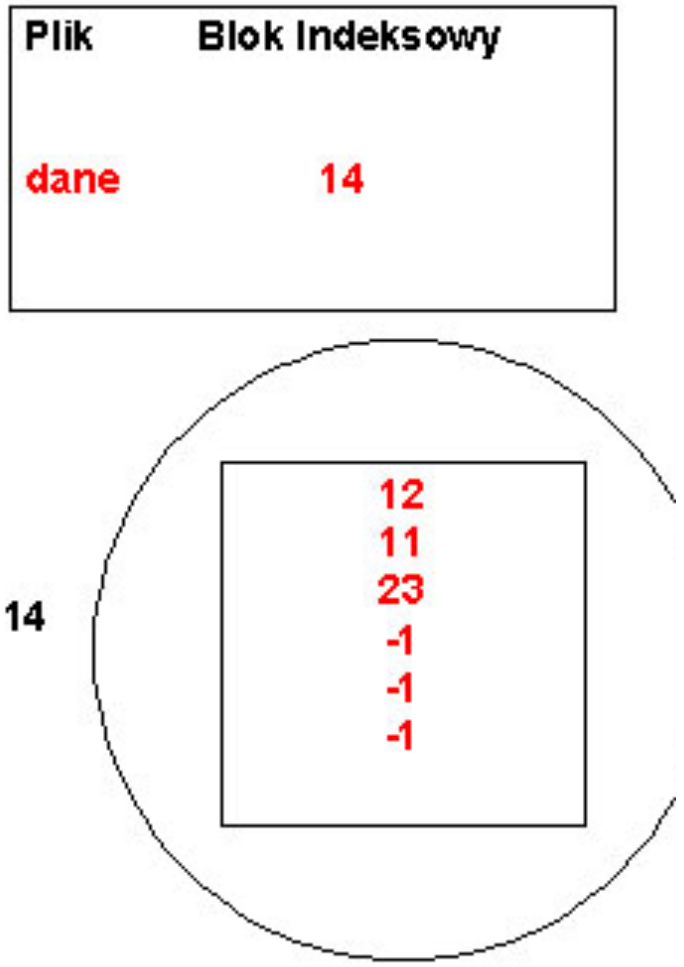


PRZYDZIAŁ INDEKSOWY

- Każdy plik ma własny blok indeksowy, będący tablicą bloków dyskowych.
- Pozycja o numerze *i* wskazuje na *i*-ty blok danych pliku.
- Deskryptor zawiera adres bloku indeksowego dla pliku.
- Aby odczytać *i*-ty blok pliku używa się *i*-tego wskaźnika z bloku indeksowego.

PRZYDZIAŁ INDEKSOWY





TWORZENIE PLIKU W PRZYDZIALE INDEKSOWYM

Przydzielony zostaje blok indeksowy.

Wszystkie wskaźniki w bloku dostają wartość pusty.

Gdy *i*-ty blok jest zapisywany po raz pierwszy, usuwa się go z listy wolnych przestrzeni, a jego adres zostaje umieszczony w pozycji o numerze *i* w bloku indeksowym.

ZALETY I WADY PRZYDZIAŁU INDEKSOWEGO

Zalety

- Umożliwia dostęp bezpośredni i sekwencyjny.
- Brak zewnętrznej fragmentacji.
- Nie trzeba określać wielkości pliku z góry.

Wady

- Marnuje miejsce na bloki indeksowe. Zużycie miejsca jest większe niż w przydziale listowym.

OKREŚLANIE WIELKOŚCI BLOKU INDEKSOWEGO

schemat listowy,

indeks wielopoziomowy,

schemat kombinowany.

SCHEMAT LISTOWY

- Blok indeksowy zawiera się zazwyczaj w jednym bloku dyskowym.
- Dla dużych plików bloki indeksowe łączy się.
- Blok indeksowy może zawierać nagłówek z nazwą pliku oraz zbiór pierwszych np.. 100 adresów bloków dyskowych. Kolejny adres będzie albo nil (dla małego pliku) albo będzie zawierał adres kolejnego bloku indeksowego.

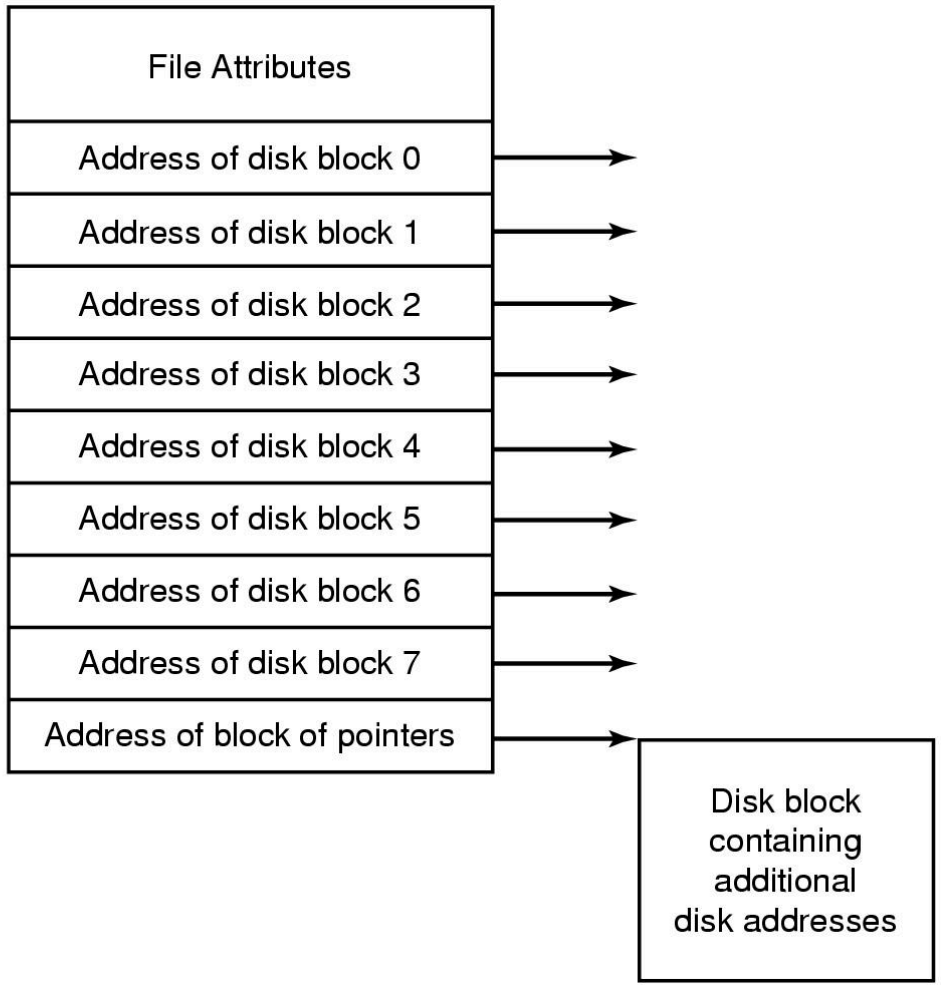
INDEKS WIELOPOZIOMOWY

- Stosuje się oddzielny blok indeksowy.
- Wskazuje on na bloki indeksowe, które zawierają wskaźniki do bloków pliku.
- Aby dojść do bloku system posługuje się indeksem pierwszego poziomu w celu znalezienia indeksu drugiego poziomu, a z niego odczyta numery bloków.
- Można stosować wskaźniki wielopoziomowe.
- Jeżeli w bloku indeksowym mieszczą się 256 wskaźników, to dwa poziomy wskaźników umożliwiają adresowanie 65536 bloków danych.

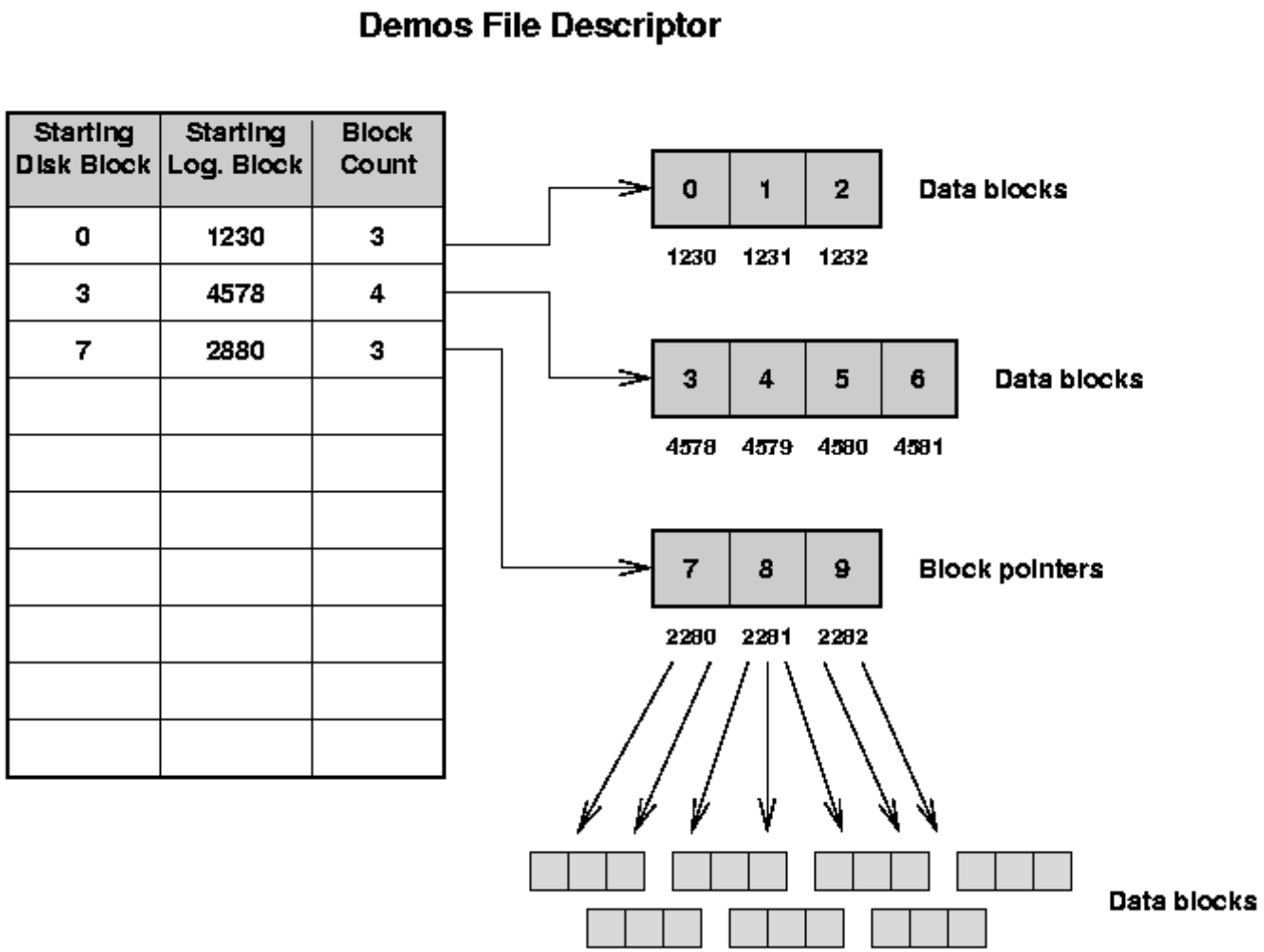
SCHEMAT KOMBINOWANY

- Schemat stosowany w UNIX BSD.
- Pewna liczba pierwszych wskaźników jest przechowywanych w katalogu urządzenia, wskazują one na *bloki bezpośrednie*.
- Bloki bezpośrednie zawierają wprost bloki z danymi. Przeznaczone są na adresowanie małych plików – zaleta: łatwy dostęp.
- Pozostałe wskaźniki wskazują na *bloki pośrednie*.
- Pierwszy wskaźnik bloku pośredniego wskazuje na blok jednokrotnie pośredni (indeks dwupoziomowy).
- Kolejny wskaźnik wskazuje na blok dwukrotnie pośredni, itd..

PRZYKŁAD I-NODE (I-WĘZŁA) BSD

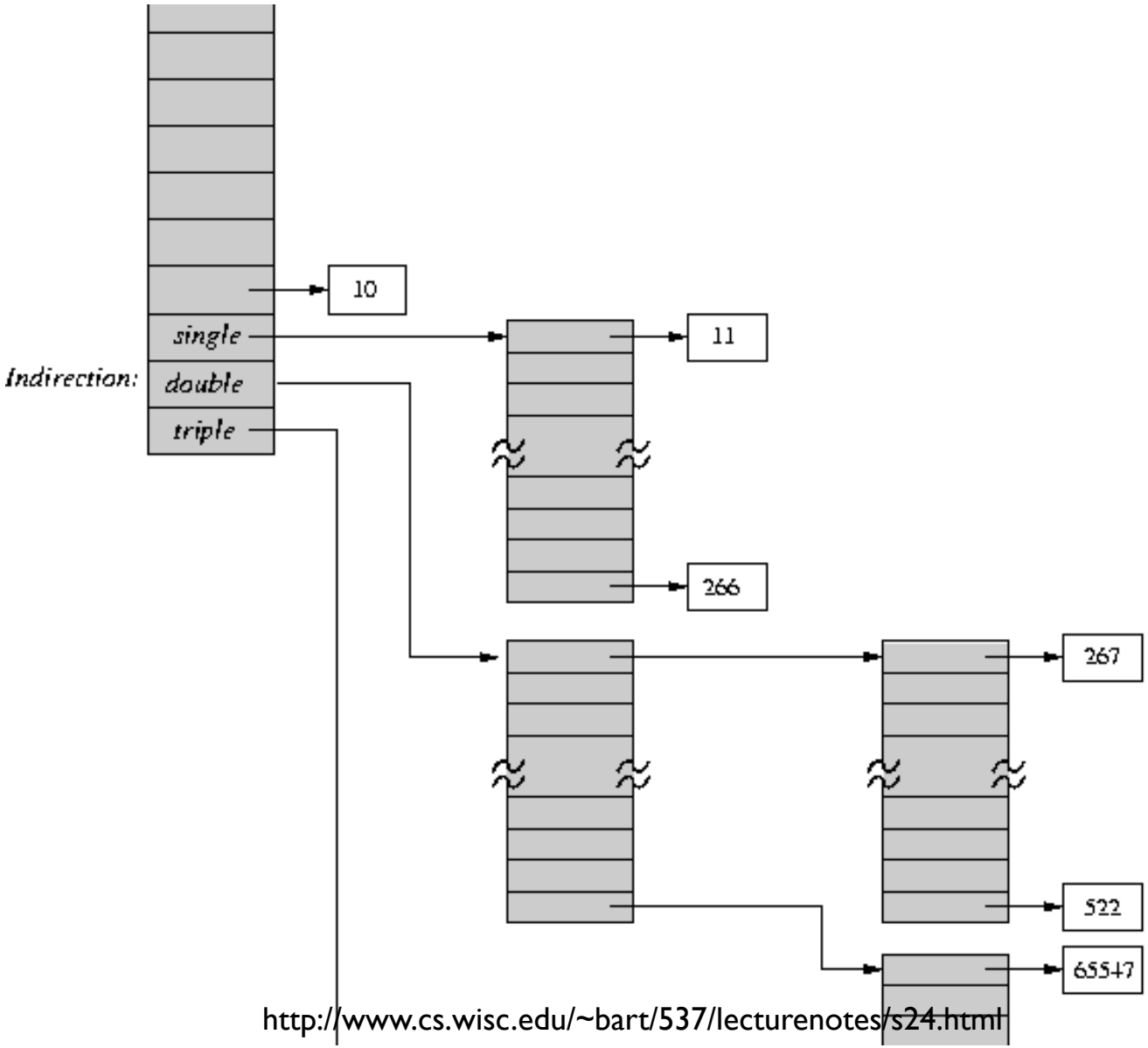


SYSTEM PLIKÓW Z CRAY I



http://www.cs.wisc.edu/~bart/537/lecturenotes/s24.html

UNIX I-NODE



http://www.cs.wisc.edu/~bart/537/lecturenotes/s24.html

KATALOGI

- Dwa cele:
  1. Zapewniają użytkownikom uporządkowany sposób przechowywania plików.
  2. Dla systemu plików, zapewniają wygodny interfejs nazewnictwa, który pozwala na implementację w celu oddzielenia struktury plików logicznych od fizycznego rozmieszczenia plików na dysku.
- Większość systemów plików obsługuje wielopoziomowe katalogi.
  - Hierarchie nazw (/, /usr, /usr/local/, ...)
- Większość systemów plików obsługuje pojęcie aktualnego katalogu.
  - Nazwy względne określone w odniesieniu do bieżącego katalogu
  - Nazwy bezwzględne zaczynają się od korzenia drzewa katalogów.

STRUKTURA KATALOGOWA

- Każdy plik zapisany na dysku musi zostać zarejestrowany w katalogu urządzenia.
- Dla złożonych systemów wymagana jest struktura katalogów.
- Struktura katalogów może obejmować różne urządzenia.
- System operacyjny przechowuje dwa katalogi:
  - urządzenia (informacje o fizycznym położeniu pliku)
  - plików (logiczna organizacja plików).

ZAWARTOŚĆ KATALOGU PLIKÓW

- Dla każdego pliku w katalogu system operacyjny przechowuje atrybuty wymienione wcześniej (slajd 20):
- Lista atrybutów jest zazwyczaj nieuporządkowana.
- Wpisy zazwyczaj posortowane przez program czytający katalogi.
- Katalogi przechowywane zazwyczaj w plikach.

PODSTAWOWE OPERACJE NA KATALOGACH

UNIX	NT
Katalogi zaimplementowane w plikach	Specjalne operacje katalogowe
Use file ops to create dirs	CreateDirectory(name) RemoveDirectory(name)
C runtime library zapewnia wyższy poziom abstrakcji do czytania katalogów	Różne metody odczytywania wpisów w katalogach
opendir(DIR) readdir(DIR) seekdir(DIR) closedir(DIR)	FindFirstFile(pattern) FindNextFile()

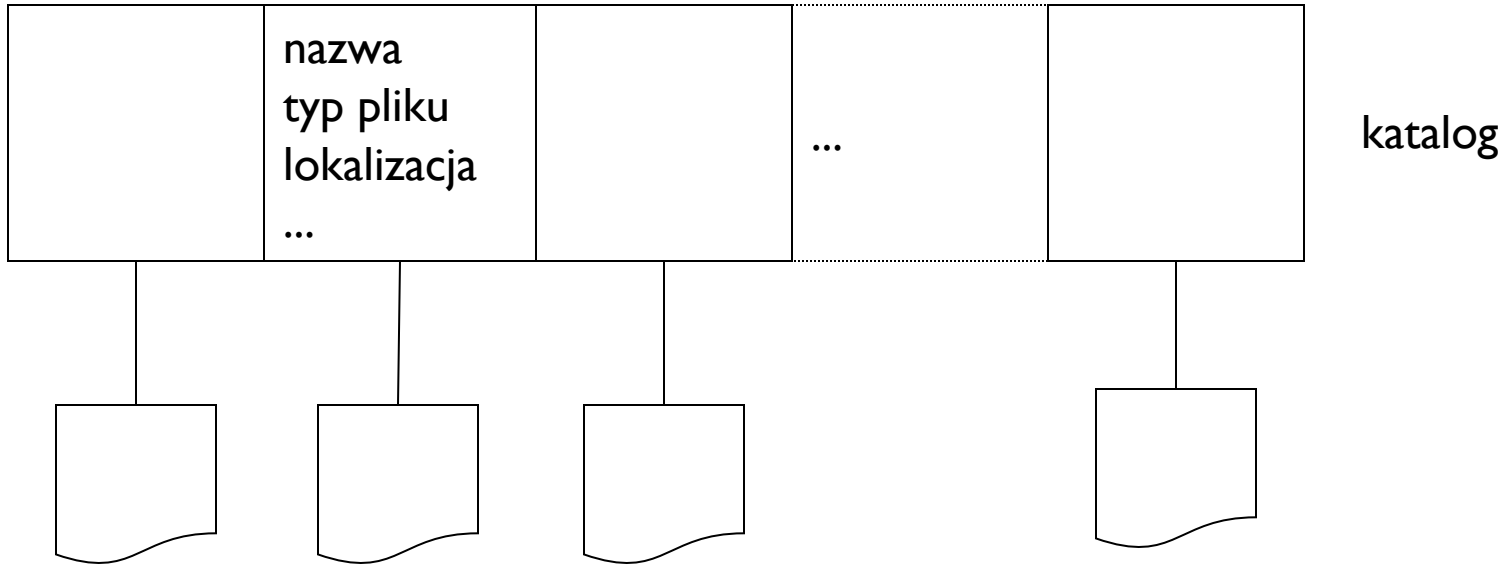
PRZEZNACZENIE SYSTEMU KATALOGOWEGO

- Użytkownik chce mieć możliwość dostępu do utworzonych plików. Może oczywiście robić to przez deskryptor pliku, ale wygodniej używać nazwy tekstowe.
- Katalog to tablica symboli przekładająca deskryptor pliku na nazwę pliku.
- Katalogi mogą być zorganizowane (na poziomie logicznym) na wiele sposobów:
  - jeden katalog dla całego dysku: jednopoziomowy,
  - katalog dwupoziomowy,
  - katalog o strukturze drzewa,
  - katalog w postaci niecyklicznego grafu.

- jeden katalog dla całego dysku,
- wydziela się specjalny obszar dysku na przechowanie katalogu,
- nazwy plików muszą być jednoznaczne,
- różni użytkownicy nie mogą używać tych samych nazw.

KATALOG JEDNOPOZIOMOWY

KATALOG JEDNOPOZIOMOWY



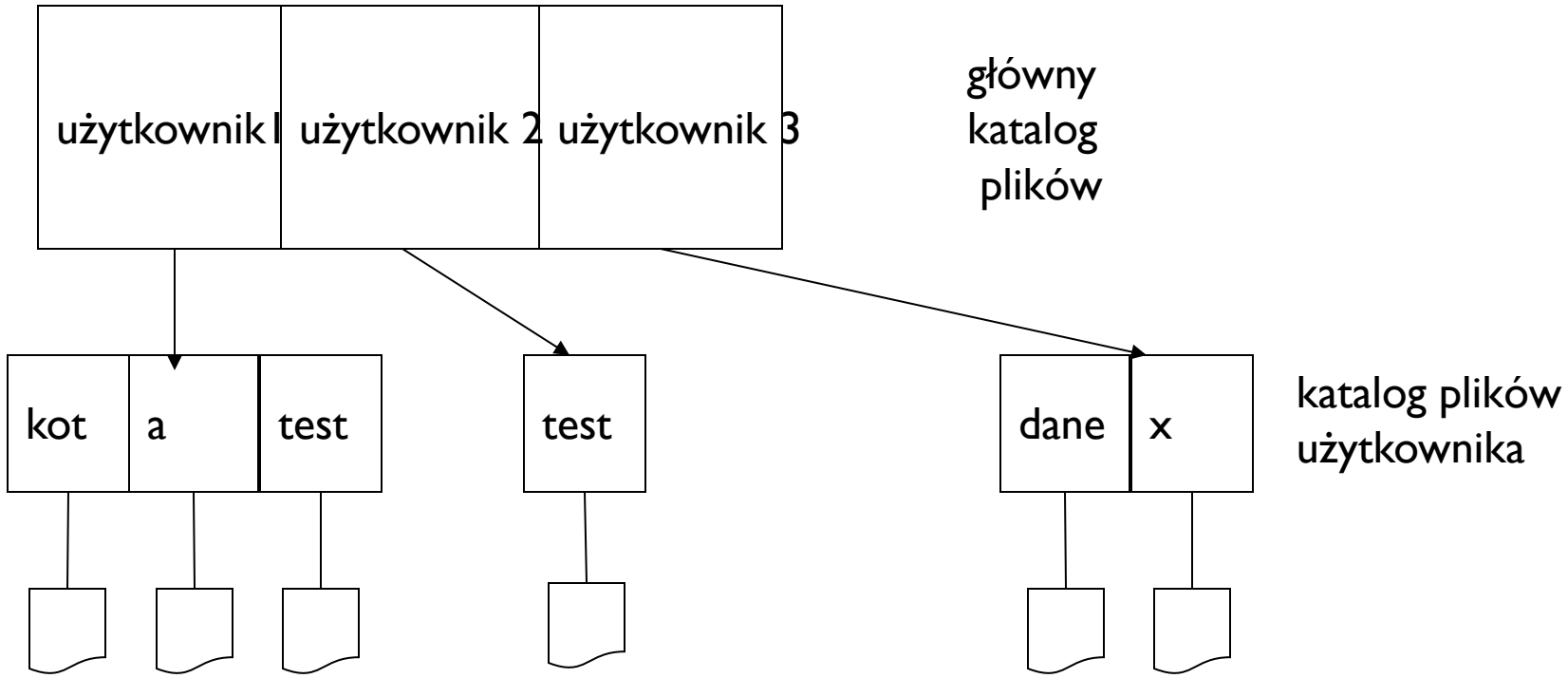
KATALOG DWUPOZIOMOWY

Każdy użytkownik ma swój katalog plików użytkownika.

Jeżeli użytkownik zarejestruje się w systemie, przeszukuje się wówczas główny katalog indeksowany nazwami użytkownika, by odnaleźć katalog użytkownika.

Przy tworzeniu, usuwaniu plików przeszukuje się tylko lokalne katalogi użytkowników.

KATALOG DWUPOZIOMOWY



KATALOG DWUPOZIOMOWY – WADY

Użytkownicy nie mogą współpracować – nie mogą nawzajem widzieć swoich katalogów.

- Rozwiązanie: stosowanie ścieżek dostępu – pozwala na jednoznaczne nazywanie pliku:
  - użytkownik b jeżeli przeszukuje swój katalog korzysta ze ścieżki np.: /usrb/plik.

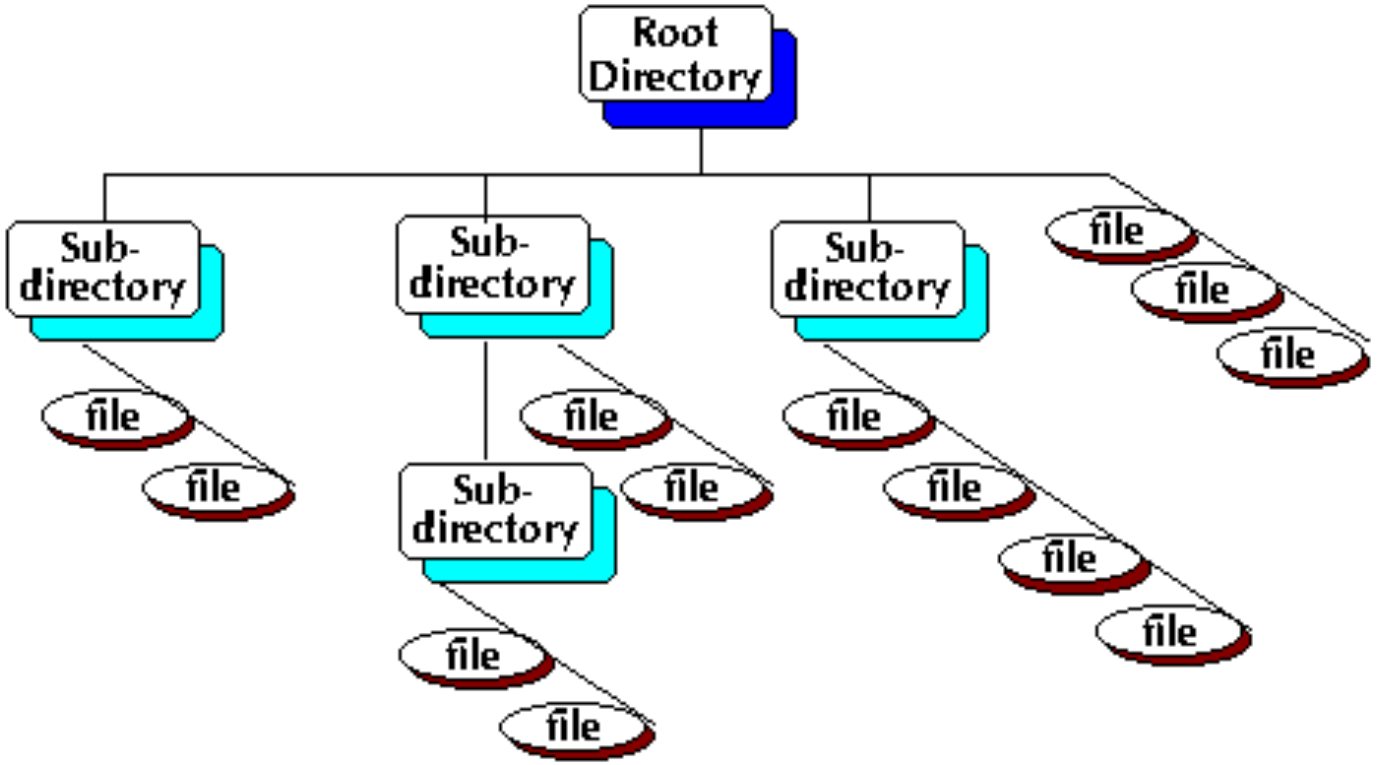
Pliki systemowe mają być dostępne dla wszystkich użytkowników. Należałoby w katalogu każdego użytkownika trzymać kopie.

- Rozwiązanie: ścieżka dostępu do wspólnego katalogu specjalnego użytkownika (sys).
  - użytkownik żąda dostępu do pliku: w pierwszej kolejności przeszuka się jego katalog, potem katalog sys.

KATALOGI O STRUKTURACH DRZEWIASTYCH

- Jest to uogólnienie katalogu dwupoziomowego i pozwala użytkownikowi na tworzenie własnych podkatalogów.
- MS-DOS ma strukturę drzewa.
- Katalog lub podkatalog zawiera zbiór plików lub podkatalogów.
- Każda pozycja w podkatalogu posiada bit odróżnienia pliku od podkatalogu.

KATALOGI O STRUKTURACH DRZEWIASTYCH

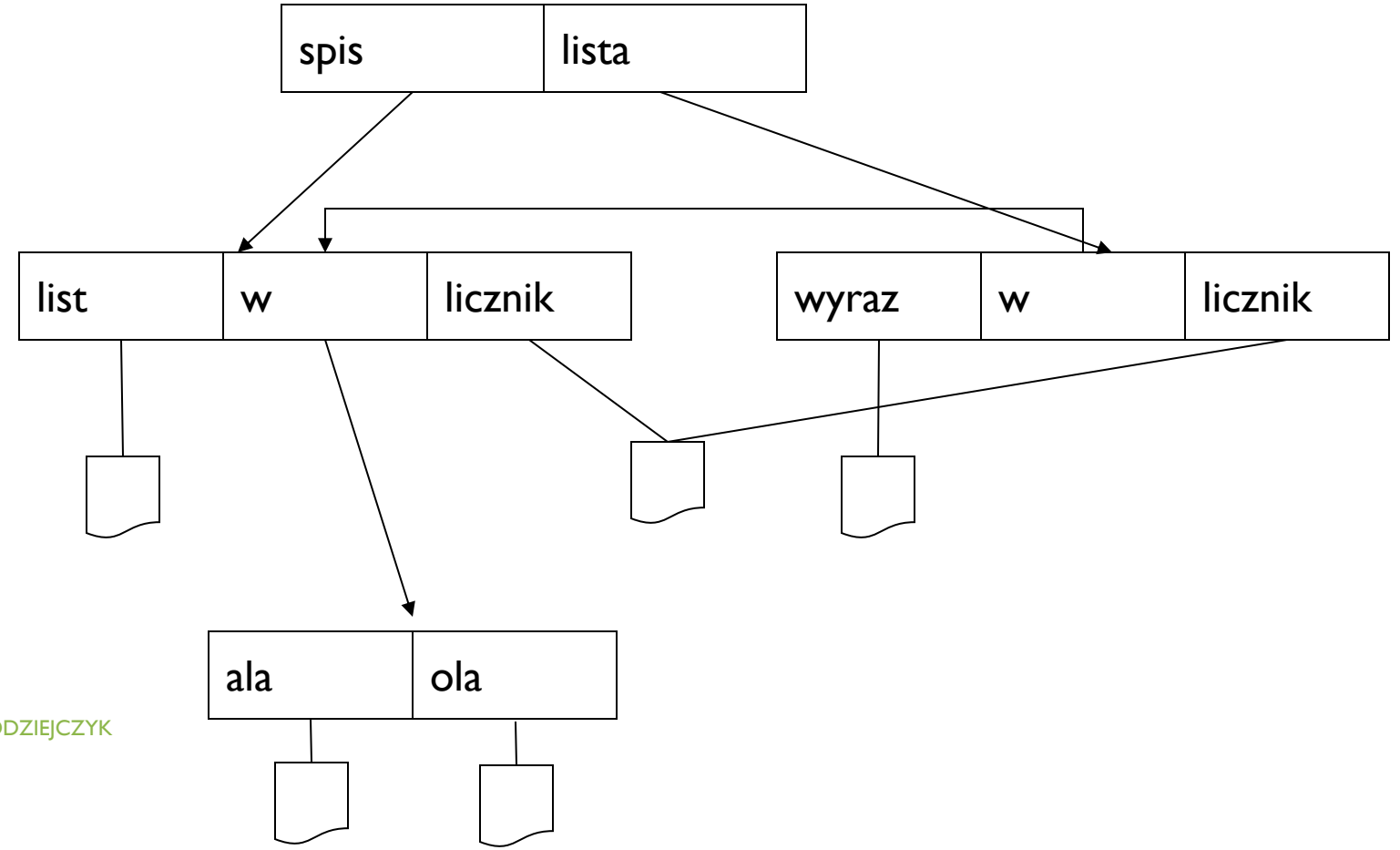


http://courses.wccnet.org/computer/mod/q02cd.htm

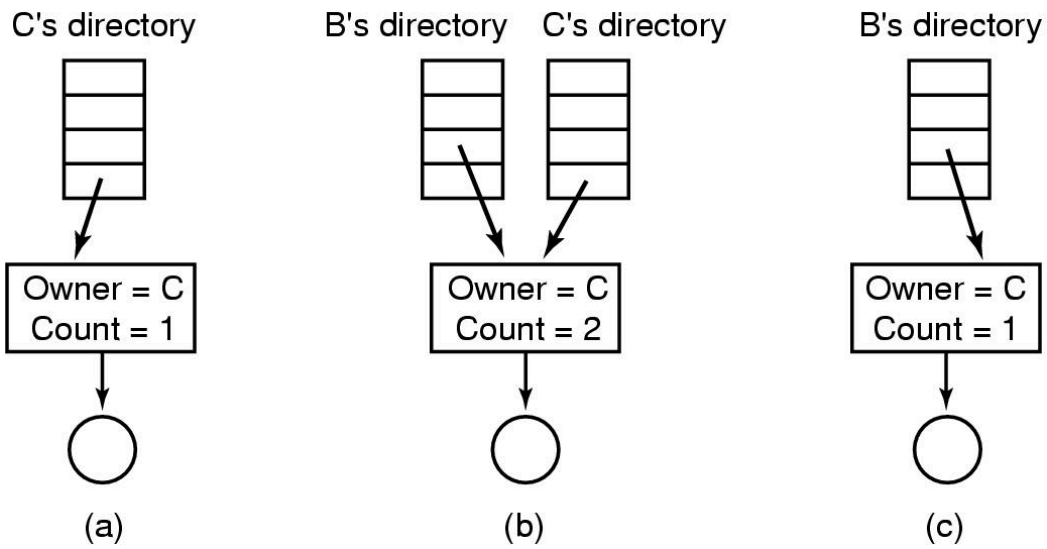
STRUKTURA GRAFU BEZ CYKLI

- Szczególnie przydatny przy pracy zespołowej.
- Nie tworzy się kopii pliku, ale plik jest współdzielony przez użytkowników.
- Fizycznie istnieje jeden plik.
- Metody implementowania katalogów i plików współdzielonych:
  - Tworzenie dowiązań - linków: jako dowiązanie może służyć bezwzględna lub względna ścieżka (link symboliczny);
  - Link twardy - podwojenie informacji w obu katalogach dzielonych: oryginał i kopia są nierozróżnialne; problemem jest utrzymanie spójności przy modyfikacji pliku.
  - Link symboliczny – zawiera tylko ścieżkę pliku.

STRUKTURA GRAFU BEZ CYKLI



PLIKI WSPÓŁDZIELONE – LINK TWARDY



- a) Sytuacja poprzedzająca powiązanie.
- b) Po utworzeniu połączenia
- (c) Po usunięciu pliku przez pierwotnego właściciela.

PROBLEMY STRUKTURY GRAFU

- Przeszukiwanie katalogu powinno unikać wielokrotnego przeszukiwania struktur dzielonych.
- Usuwanie pliku:
  - problem: pozostawienie wskaźników do nieistniejącego pliku (lub z czasem do zupełnie innego pliku, gdy przestrzeń zostanie ponownie zapisana) [nie dotyczy dowiązań symbolicznych].
  - można plik pozostawić tak długo, aż nie zostaną usunięte wszystkie do niego dowiązania. Można to zrealizować poprzez listę zarejestrowanych dowiązań, lub licznika dowiązań (UNIX).

OPERACJE NA KATALOGACH

- Create - Tworzony jest katalog. Jest on pusty, za wyjątkiem kropki i dotdot, które są umieszczane automatycznie przez system.
- Delete - Katalog jest usuwany. Można usunąć tylko te katalogi, które są puste.
- Opendir - Katalogi można czytać. Ale przed odczytaniem jakiegokolwiek katalogu, musi on zostać najpierw otwarty. Dlatego, aby wyświetlić listę wszystkich plików znajdujących się w danym katalogu, program wyświetlający listę otwiera ten katalog, który wymaga odczytania nazwy wszystkich plików, które ten katalog zawiera.
- Closedir - Katalog powinien być zamknięty tylko po to, aby zwolnić wewnętrzną przestrzeń w tablicy po jego przeczytaniu.
- Readdir - To wywołanie zwraca następny wpis w otwartym katalogu.
- Rename - Katalog można również zmienić nazwę, tak jak pliki.
- Link - Linkowanie jest techniką, która pozwala na pojawienie się pliku w więcej niż jednym katalogu.
- Uplink - Wpis do katalogu jest usuwany.

TRANSLACJA ŚCIEŻKI

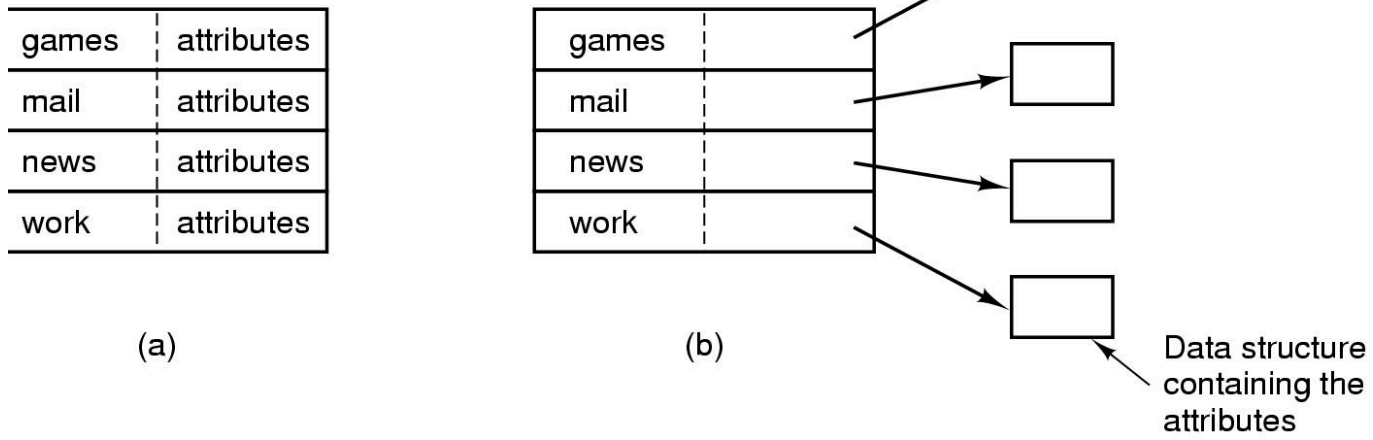
- Powiedzmy, że chcesz otworzyć "/one/two/three"
- Co robi system plików?
  - Open directory "/" (well known, can always find)
  - Search for the entry "one", get location of "one" (in dir entry)
  - Open directory "one", search for "two", get location of "two"
  - Open directory "two", search for "three", get location of "three"
  - Open file "three"
- Systemy spędzają dużo czasu na chodzeniu po ścieżkach katalogowych.
  - Dlatego open jest oddzielony od odczytu/zapisu.
  - System operacyjny będzie buforował prefiksy wyszukiwania pod kątem wydajności.

IMPLEMENTACJA KATALOGÓW

Limitowana nazwa długości pliku np. 255 znaków

a) Prosty katalog zawierający wpisy o stałej wielkości z adresami dysków i atrybutami we wpisie do katalogu.

b) Katalog, w którym każdy wpis odnosi się tylko do węzła i.

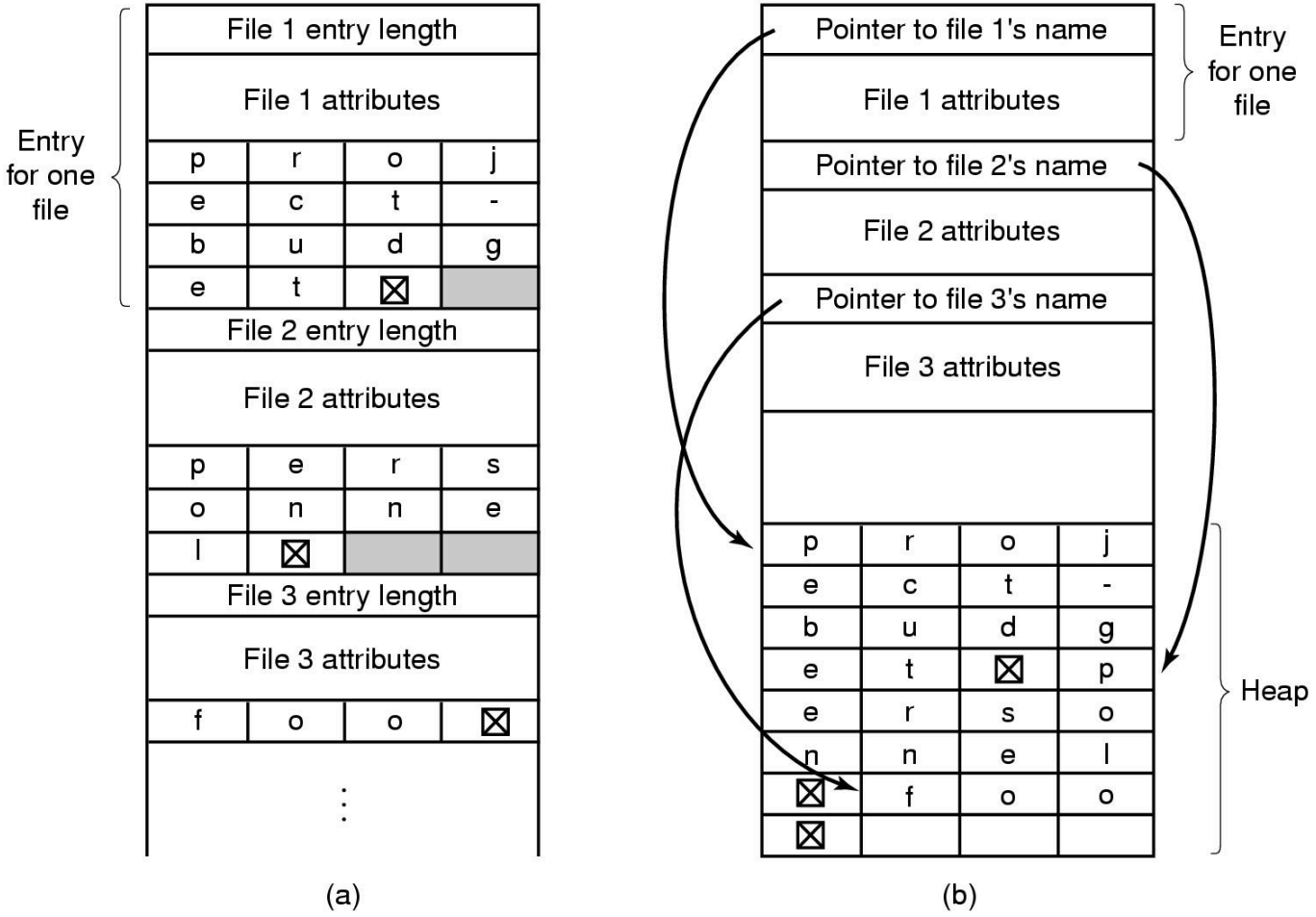


IMPLEMENTACJA KATALOGÓW

Dwa sposoby postępowania z długimi nazwami plików w katalogu.

a) In-line stała długość słowa – po usunięciu pliku w katalogu dziura o nieregularnym rozmiarze

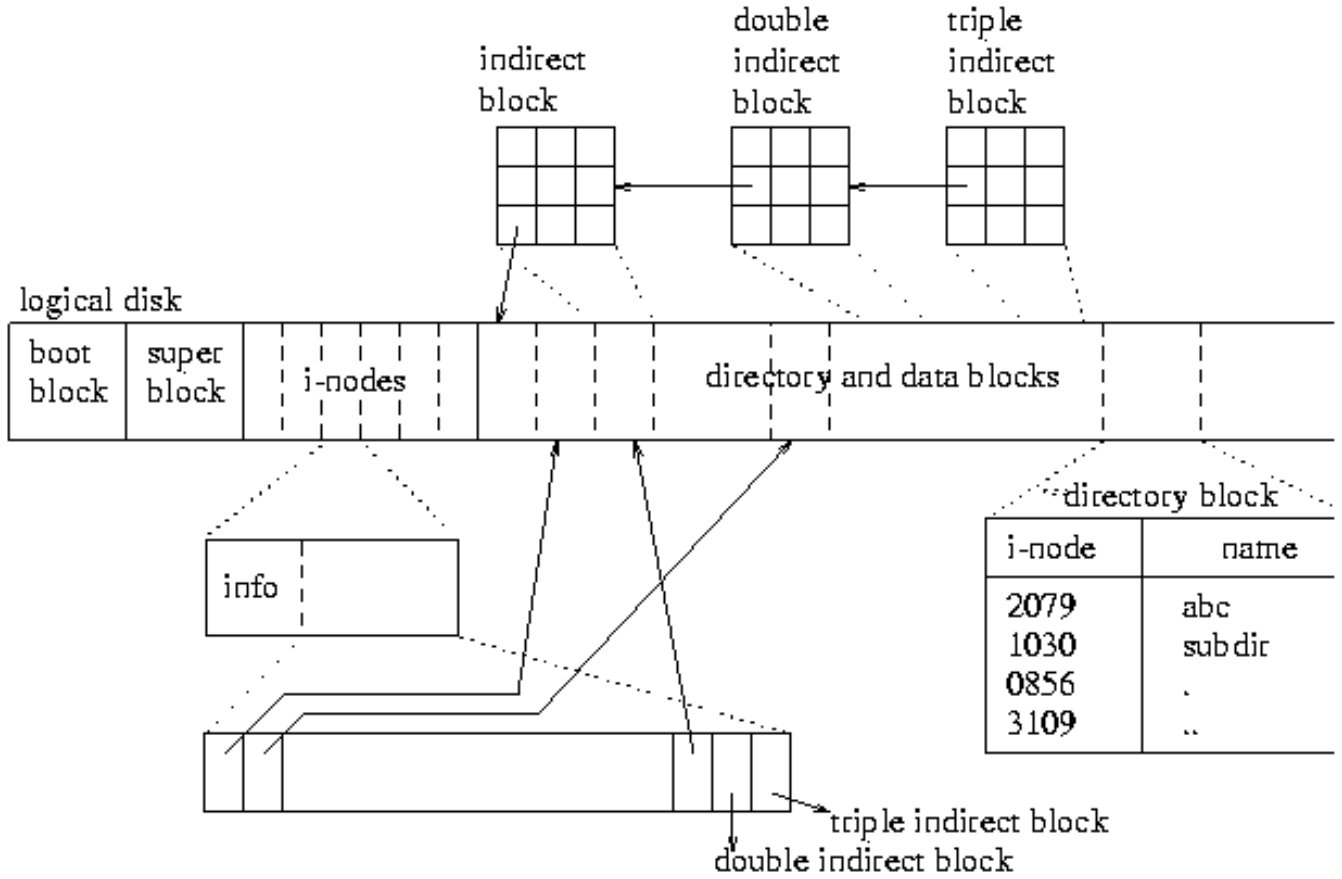
b) Na stosie – na końcu katalogu



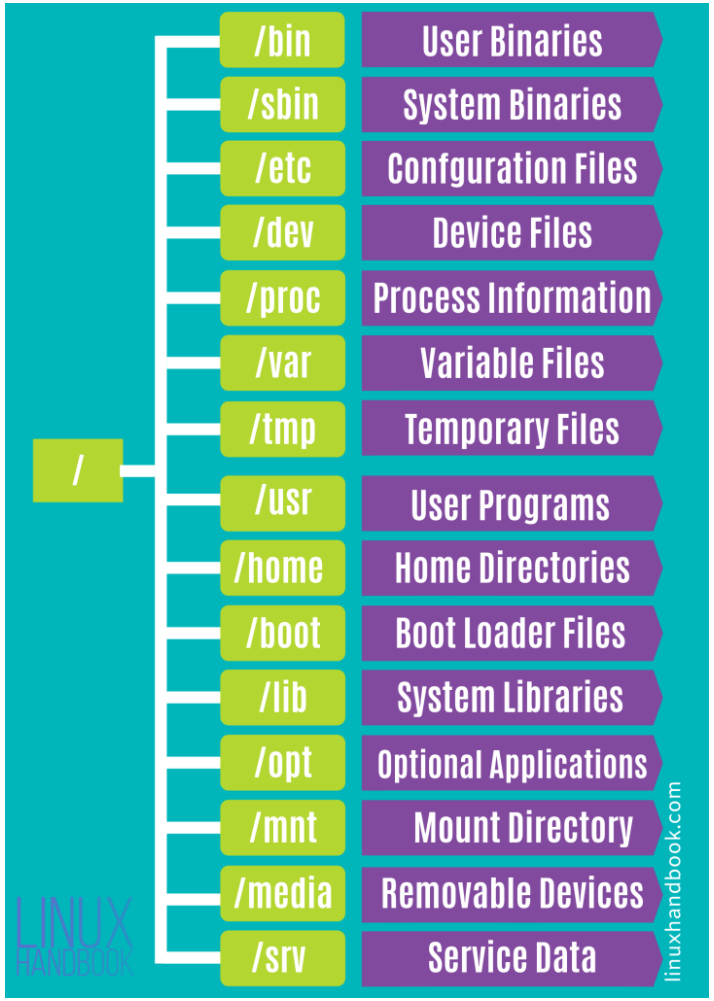
UNIX

- Wszystkie deskryptory trzymane są w tablicy określonego rozmiaru. Deskryptor zawiera informacje o ochronie.
- Przeznaczona jest na to specjalna część dysku. Dysk jest podzielony na dwie części: tablica z deskryptorami i reszta, czyli bloki dla danych i bloki pośrednie.
- Wielkość tablicy deskryptorów określania jest podczas tworzenia systemu plików i jest niezmienna.
- W Unix, deskryptor nazywa się *i-node*, a jego indeks w tablicy nazywa się *i-number*. System stosuje i-number w odniesieniu do pliku (nie posługuje się nazwami).
- Jeżeli plik jest otwarty to i-node jest przechowywany w pamięci głównej. Po zamknięciu pliku i-node jest zrzucany na dysk.

UNIX


indirect block
double indirect block
triple indirect block
logical disk
boot block
super block
i-nodes
directory and data blocks
info
directory block
i-node
name
2079
abc
1030
subdir
0856
.
3109
..
triple indirect block
double indirect block

SYSTEM KATALOGÓW W UNIX



http://www.cs.wisc.edu/~bart/537/lecturenotes/s25.html

KATALOGI W UNIX

- Katalogi przechowywane są na dysku jako zwykłe pliki (tj. zwykły i-node). Programy czytają katalogi jak inne pliki. Do katalogów mogą pisać tylko specjalne funkcje systemowe.
- Wskaźnik wskazany przez indeks może być innym katalogiem i w ten sposób tworzy się strukturę hierarchiczną.
- Jeden z katalogów jest katalogiem specjalnym i nazywa się go root lub korzeń. Nie ma on nazwy i jest plikiem wskazanym przez i-node 2. (0 i 1 zarezerwowane dla innych celów).

SYSTEM EXT2

drugi rozszerzony system plików (ext2),

obsługuje partycje o rozmiarze do 2 GB, nazwy plików o długości do 255 znaków,

zawiera mechanizm zapobiegający znacznej fragmentacji dysku,

rozpoznanie uszkodzonych plików następuje już przy starcie systemu,

utracone sektory dysku zapisywane są w katalogu lost+found .

SYSTEM EXT3

Przykład systemu plików z księgowaniem (ang. journaling),

Większość systemów plików z księgowaniem przechowuje w dzienniku tylko różnice pomiędzy blokami danych do zapisania i tymi, które są zapisane na dysku (listę bajtów do zmiany), co nazywamy księgowaniem logicznym (*logical journaling*).

W dzienniku przechowywane są całe zmienione bloki. Takie podejście nazywa się księgowaniem fizycznym (*physical journaling*). W razie nagłego wyłączenia systemu umożliwia on odbudowanie integralności systemu plików.

Przykład: po awarii zasilania serwer z kilkoma dużymi dyskami z ext2 potrzebuje ok. 10 min na odbudowanie spójności systemu plików; po zamianie na ext3 ta sama operacja trwa 3 sekundy.

FAT32

Dostęp jest możliwy tylko za pośrednictwem systemu Windows 95 OSR2, Windows 98, Windows Millennium Edition, Windows 2000 i Windows XP.

Woluminy mogą mieć rozmiar od 512 MB do 2 TB.

W systemie Windows XP można formatować wolumin FAT32 o maksymalnym rozmiarze 32 GB.

Maksymalny rozmiar pliku wynosi 4 GB.

NTFS

- Komputer z systemem Windows obsługuje system NTFS.
- Zalecanym minimalnym rozmiarem woluminu jest 10 megabajtów (MB).
- Możliwe są partycji znacznie większe niż 2 terabajty (TB).
- Rozmiar pliku jest ograniczony tylko rozmiarem partycji.

AUTOR: DR. INŻ JOANNA KOŁODZIEJCZYK

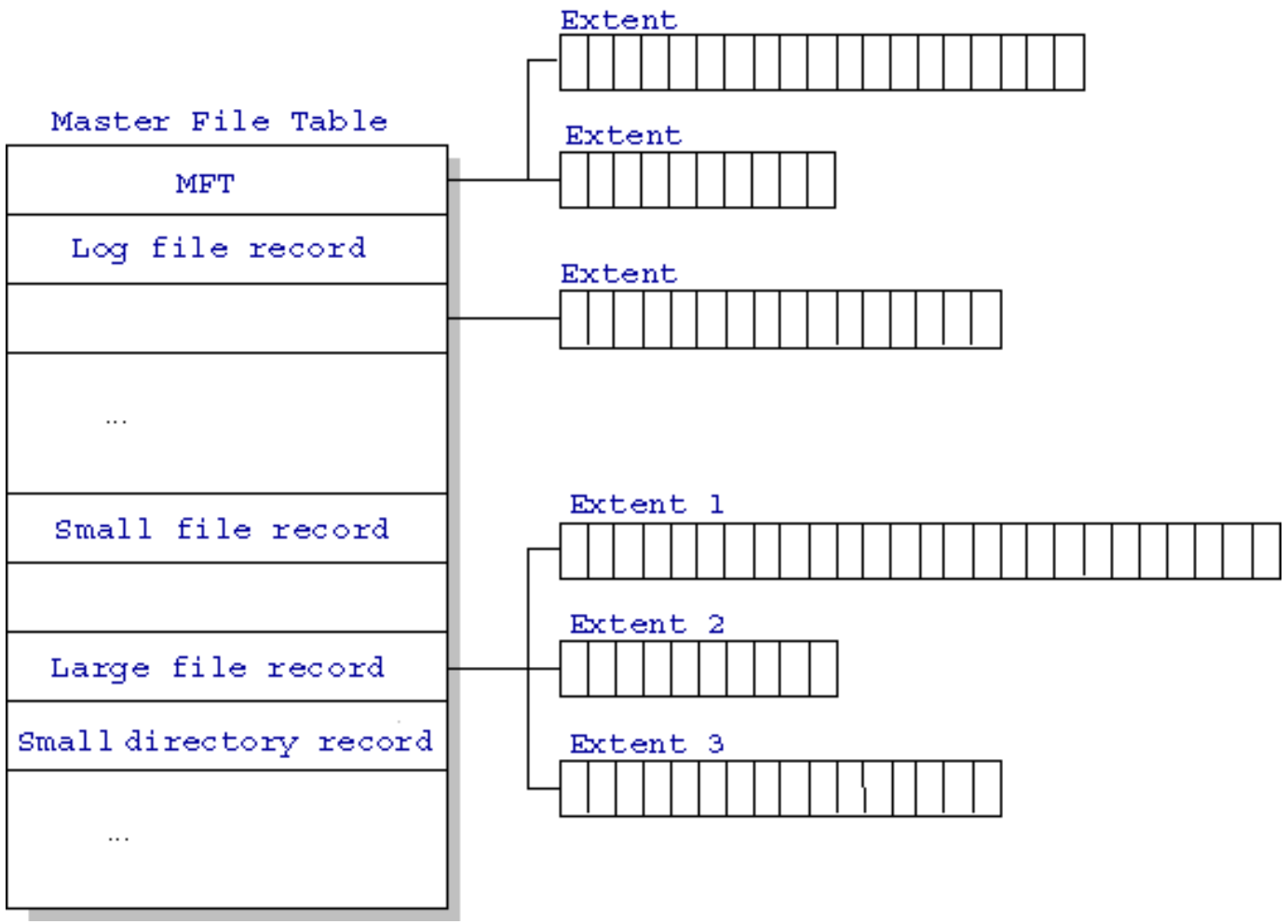
NTFS

- Jest podobny do FAT tj. pliki są opisane w pojedynczej tablicy zwanej *Master File Table* (MFT). Jakkolwiek ma bardziej nowoczesną charakterystykę, ponieważ wszystkie komponenty są plikami, włączając:
  - Master File Table,
  - pliki danych,
  - katalogi,
  - boot images,
  - logi do odzyskiwania.

MFT

- Wszystkie pliki (obiekty przechowywane na dysku) są opisane w MFT.
- MFT można opisać jako tablicę, w której każdy wiersz przeznaczony jest na plik.
- Pierwszy rekord tej tabeli opisuje samą tabelę plików podstawowych, po którym następuje rekord lustrzany MFT.
- Jeżeli pierwszy rekord MFT jest uszkodzony, NTFS odczytuje drugi rekord w celu znalezienia pliku lustrzanego MFT, którego pierwszy rekord jest identyczny z pierwszym rekordem MFT. Lokalizacje segmentów danych zarówno dla pliku lustrzanego MFT, jak i MFT są zapisywane w sektorze startowym.
- MFT przydziela określoną ilość miejsca dla każdego rekordu pliku. Atrybuty pliku są zapisywane na przydzielonym miejscu w pliku MFT. Małe pliki i katalogi (zazwyczaj 512 bajtów lub mniejsze), mogą być w całości zawarte w rekordzie głównym tabeli plików.

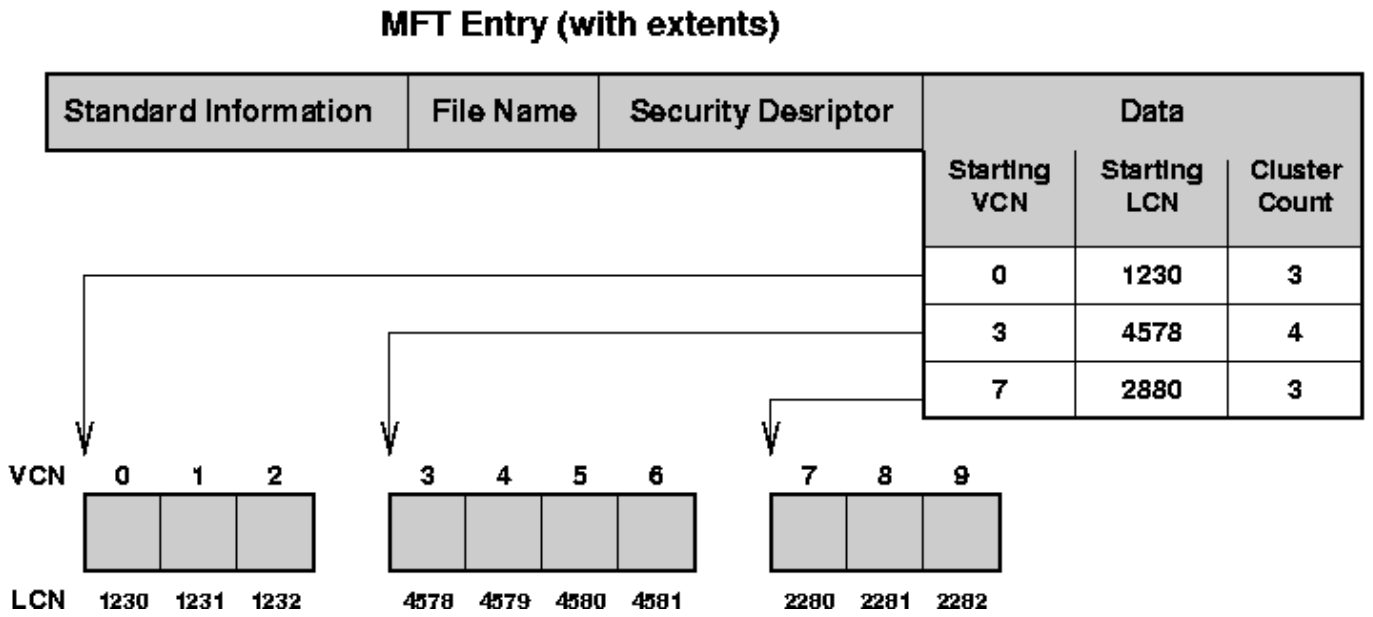
MFT W NTFS



http://www.ntfs.com/ntfs-mft.htm

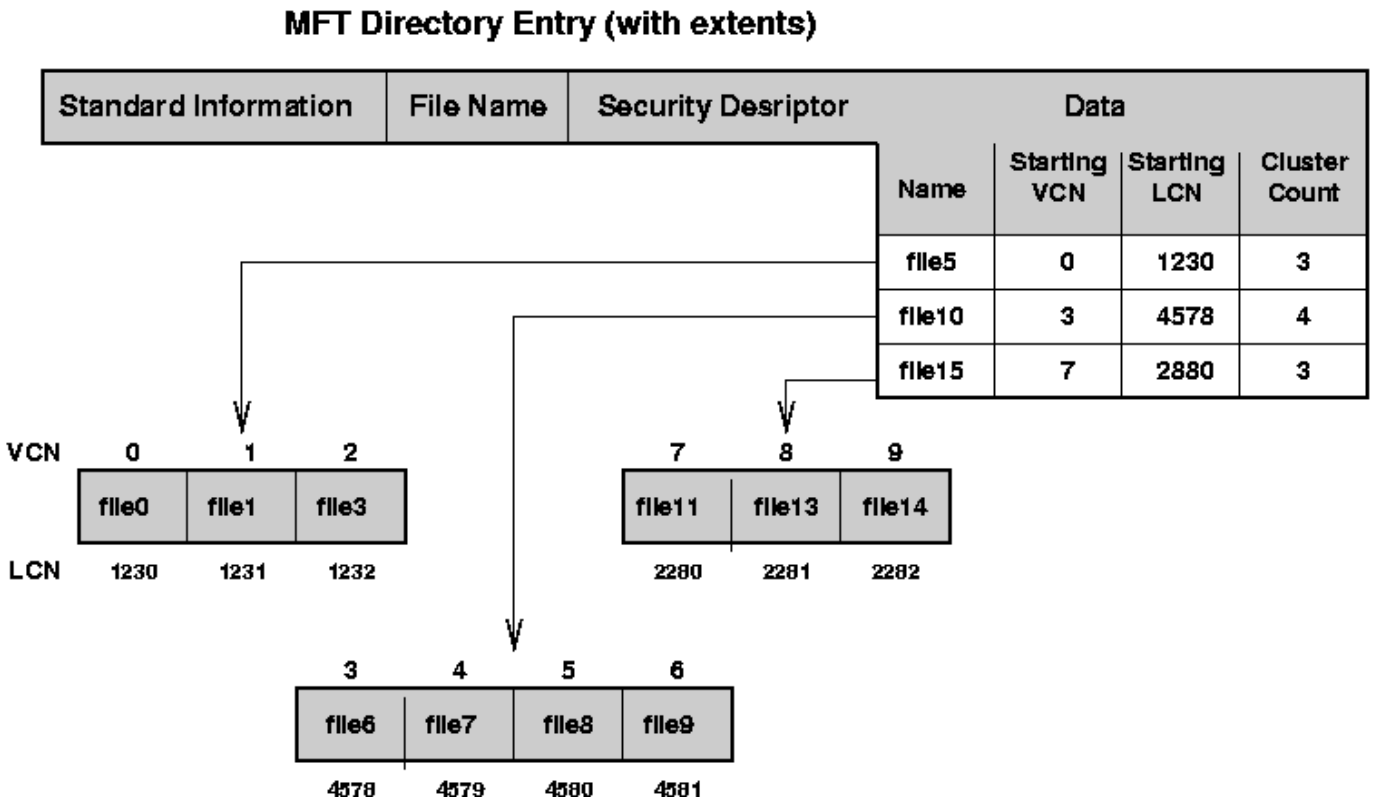
POLE DANYCH DLA PLIKU W MFT

- *virtual cluster number* (VCN) – numer klastru wirtualnego
- *logical cluster number* (LCN) – numer klastru logicznego



http://www.cs.wisc.edu/~bart/537/lecturenotes/s26.html

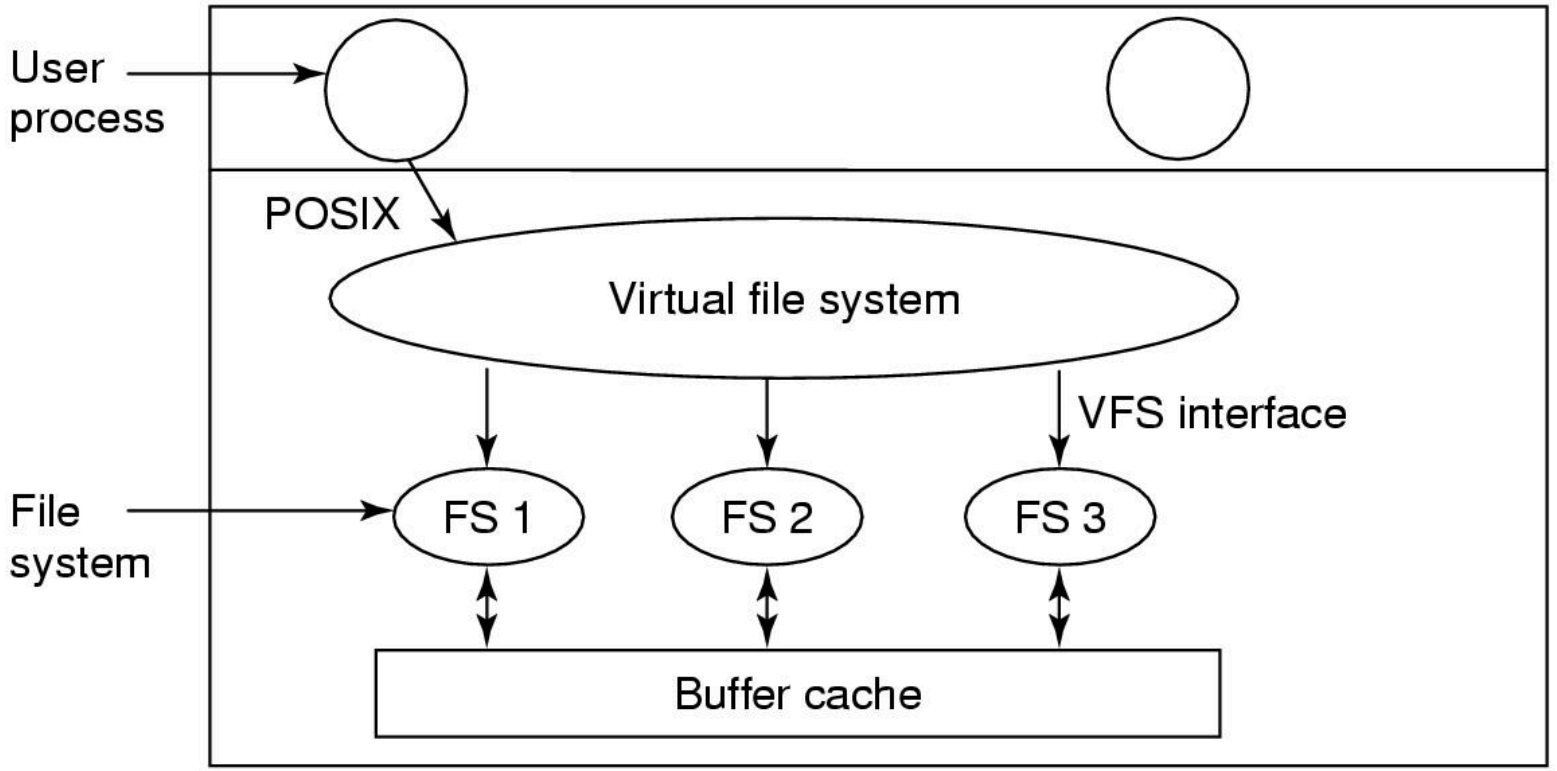
POLE DANYCH DLA KATALOGU W MFT



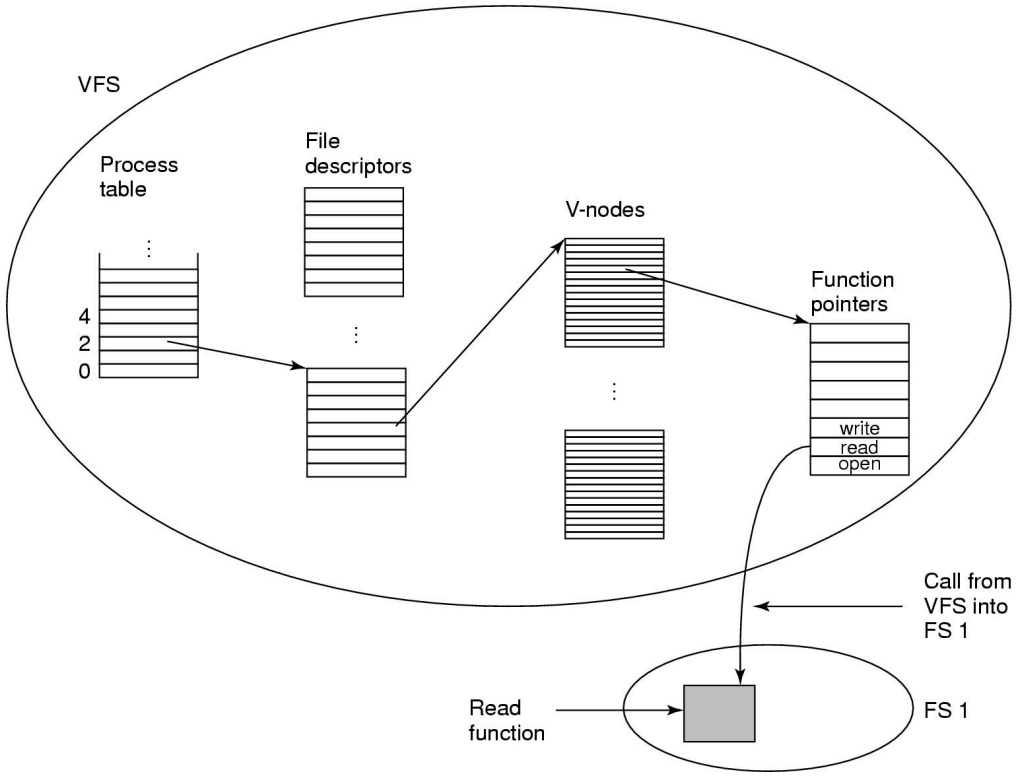
PORÓWNANIE SYSTEMÓW FAT, FAT32 I NTFS

- NTFS zapewnia kontrolę dostępu do plików i folderów oraz obsługę ograniczonego konta. System FAT32 nie ogranicza dostępu niezależnie od typu konta.
- System NTFS jest systemem plików, który najlepiej (z Windowsowych) działa z dużymi dyskami.
- Po wykonaniu konwersji dysku lub partycji na system NTFS, nie można po prostu wykonać konwersji na system FAT lub FAT32. W takim wypadku jest konieczne ponowne formatowanie dysku lub partycji.

WIRTUALNE SYSTEMY PLIKÓW



UPROSZCZONY WIDOK STRUKTUR DANYCH I KODU UŻYWANY PRZEZ VFS I KONKRETNY SYSTEM PLIKÓW DO ODCZYTU



DYSK I JEGO PODZIAŁ

- Każdy obszar jest nazwany partycją, woluminem.
- Każda partycja może mieć inny system plików taki jak: NTFS, FAT lub wybrany Unixowy.
- Z racji tego, że każda partycja może mieć własny system plików, każda z nich posiada swój katalog root.
- Wiele partycji pozwala na ograniczenie rozmiaru szczególnych części drzewa np.. ograniczając rozmiar katalogów tymczasowych.
- Wiele partycji pozwala jednemu dyskowi zachowywać się jak wiele dysków z niezależnymi bootowalnymi systemami operacyjnymi.