Sisteme de Operare

- Planificarea accesului la disc
- Algoritmi de planificare a accesului la disc
- Managementul swap-ului
- Studii de caz

Planificarea accesului la disc

- Pentru a putea avea acces la un anumit sector de disc, unitatea de disc trebuie să acţioneze în trei etape:
- poziţionarea furcii pe care se află capetele de citire pe cilindrul care conţine sectorul dorit;
- aşteptarea rotaţiei discului până când sectorul trece prin faţa capului de citire/scriere;
- schimbul propriu-zis de informaţie.

Planificarea accesului la disc

- Operaţia de poziţionare este cea mai mare consumatoare de timp
- există o coadă de aşteptare pentru accese la disc, partajată între mai mulţi utilizatori:
  - Ordinea de deservire este decisă de către SO, astfel încât serviciile aceluiaşi utilizator să se facă în ordinea sosirii lor
  - În acelaşi timp, SO trebuie să planifice serviciile oferite utilizatorilor, astfel încât timpul total de poziţionare să fie cât mai mic.
- timpului necesar transferului de informaţie nu se ia în considerare, deoarece aceasta este o constantă de construcţie a discului.

Planificarea accesului la disc

- Tehnicile folosite pentru îmbunătăţirea timpului de acces la disc variază în funcţie de producător.
- Sistemul de operare poate beneficia de acestea, folosind drivere care îi permit gestiunea simultană a mai multor discuri, care realizează o suprapunere a operaţiilor de poziţionare la mai multe discuri simultan.
- Deşi sectoarele de pe pistele exterioare sunt mai lungi decât cele de pe pistele interioare, volumul de informaţie este acelaşi, lucru folosit de producătorii de discuri pentru a îmbunătăţi performanţele.
- Din punctul de vedere al SO sunt importante metodele de reducere a timpului de acces la disc, de aşteptare a rotaţiei şi a timpului de poziţionare.

Reducerea timpului de aşteptare a rotației

- se poate face prin ordonarea adecvată a cererilor de acces la disc existente la un moment dat pentru acelaşi cilindru.
- Cel mai folosit algoritm este SLTF (Short Latency Time First) – procesul care aşteaptă cel mai puţin va fi servit primul

Reducerea timpului de poziționare

- Există o serie de elemente care nu ţin de planificare şi care pot influenţa randamentul discului:
  - structura informaţiei pe disc influenţează timpul de poziţionare.
  - Astfel, pentru un fişier secvenţial, alocat într-o zonă contiguă pe disc, timpul de acces este mult mai mic decât cel de la un fişier secvenţial - indexat.

Algoritmi de planificare a accesului la disc

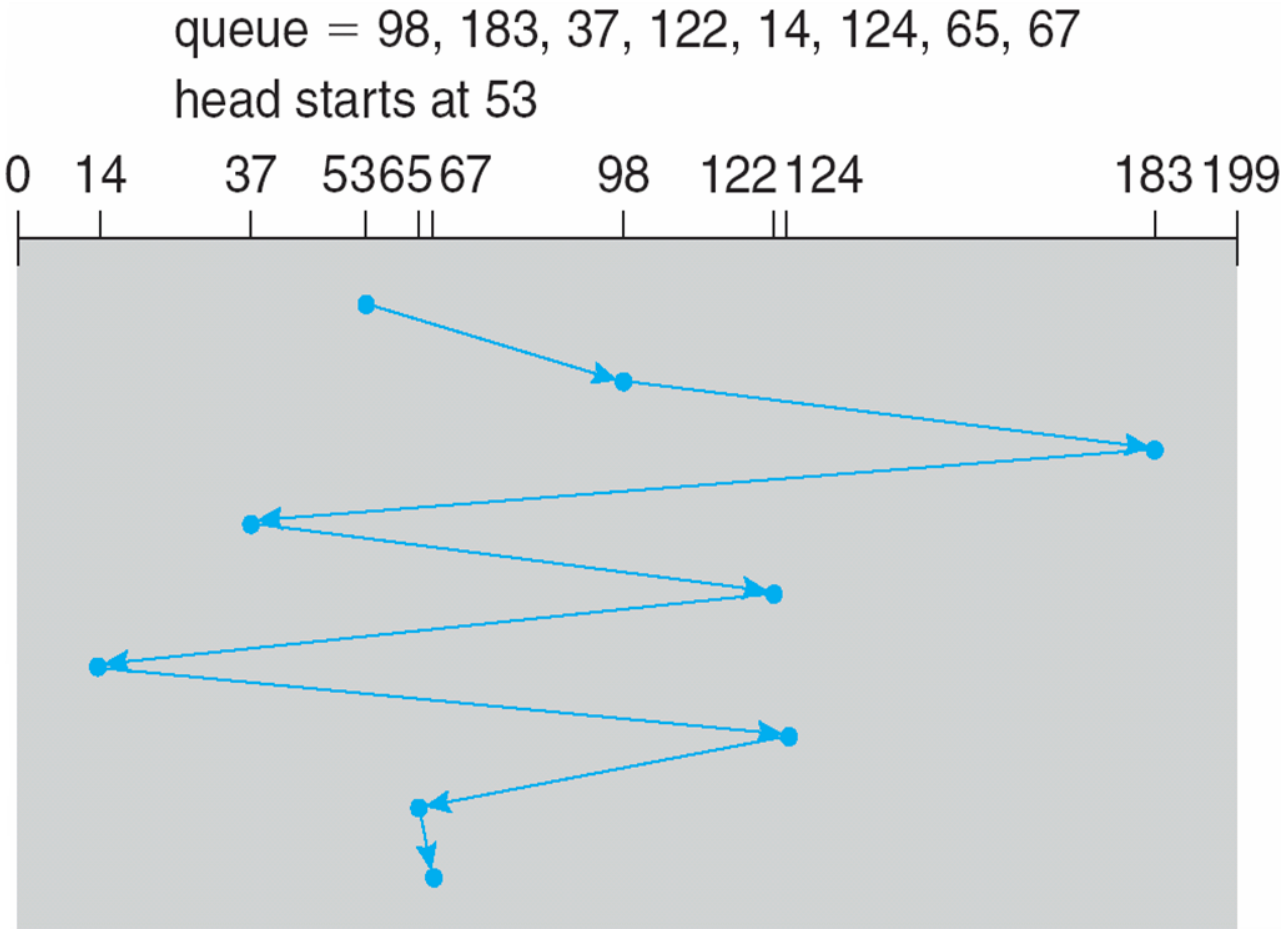
- Aceşti algoritmi se referă la planificarea poziţionării capetelor de citire/scriere a discului.
- Pentru exemplificare, vom considera următoarea secvenţă de cereri de poziţionare pe următorii cilindri: 98, 183, 37, 122, 14, 124, 65, 67 şi vom presupune că avem capetele poziţionate pe cilindrul 53 (discul are 199 cilindri).

Algoritmi de planificare a accesului la disc

- FCFS (First Come First Served)
  - **execută cererile în ordinea în care au venit.**
  - Deşi este simplu, algoritmul nu este foarte eficient, deoarece sunt necesare mişcări de poziţionare însumând 640 cilindri pentru secvenţa de cereri de mai sus.

Algoritmi de planificare a accesului la disc

- FCFS (First Come First Served)

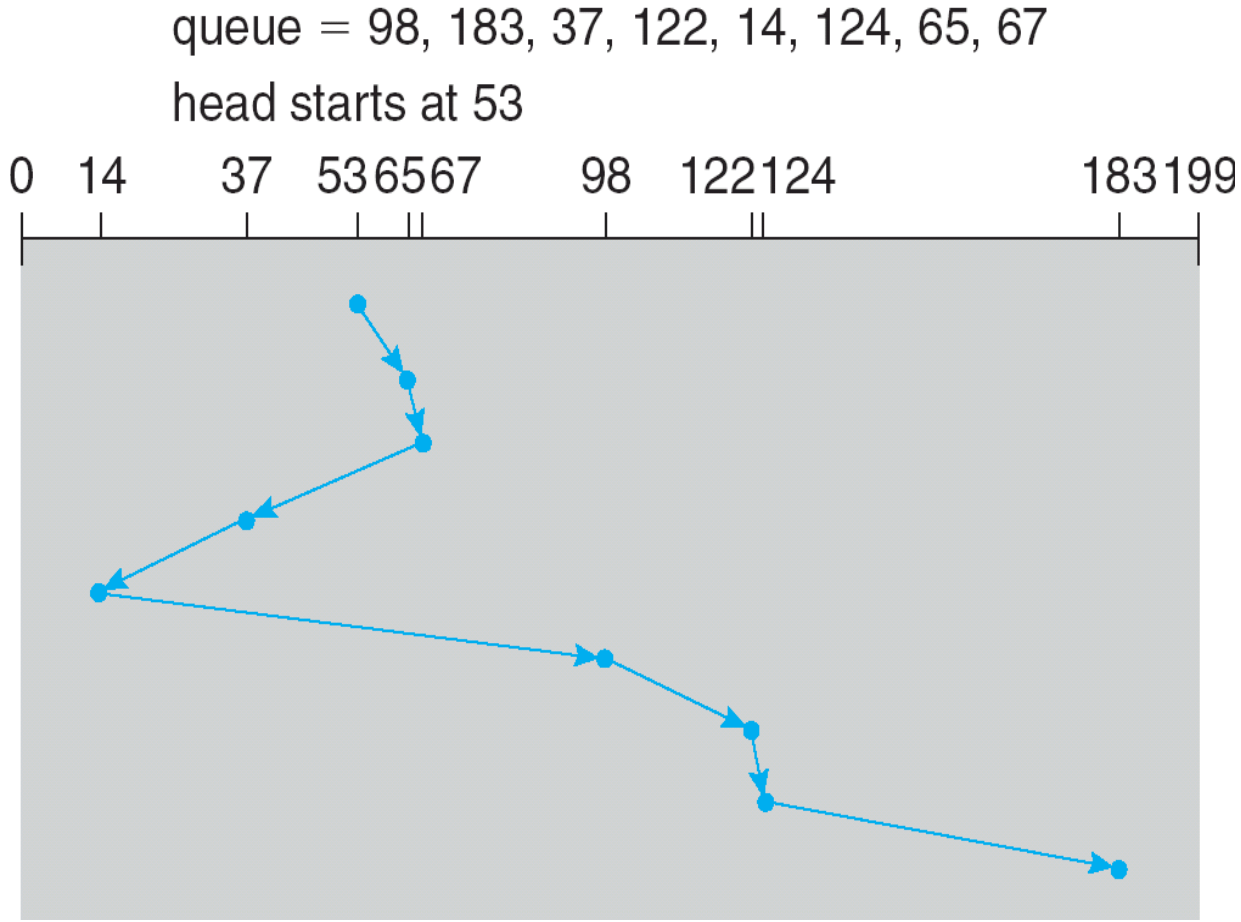


Algoritmi de planificare a accesului la disc

- SSTF (Shortest Seek Time First)
  - **execută de fiecare dată operaţia care solicită cea mai scurtă poziţionare, faţă de locul curent.**
  - Pentru secvenţa de cereri luată ca exemplu, avem următoarea ordine de deservire: 65, 67, 37, 14, 98, 122, 124, 183.
  - Această ordine necesită poziţionări peste numai 236 cilindri.
  - Principalul dezavantaj este acela că poate amâna indefinit anumite cereri deoarece coada de aşteptare se modifică permanent.

Algoritmi de planificare a accesului la disc

- SSTF (Shortest Seek Time First)

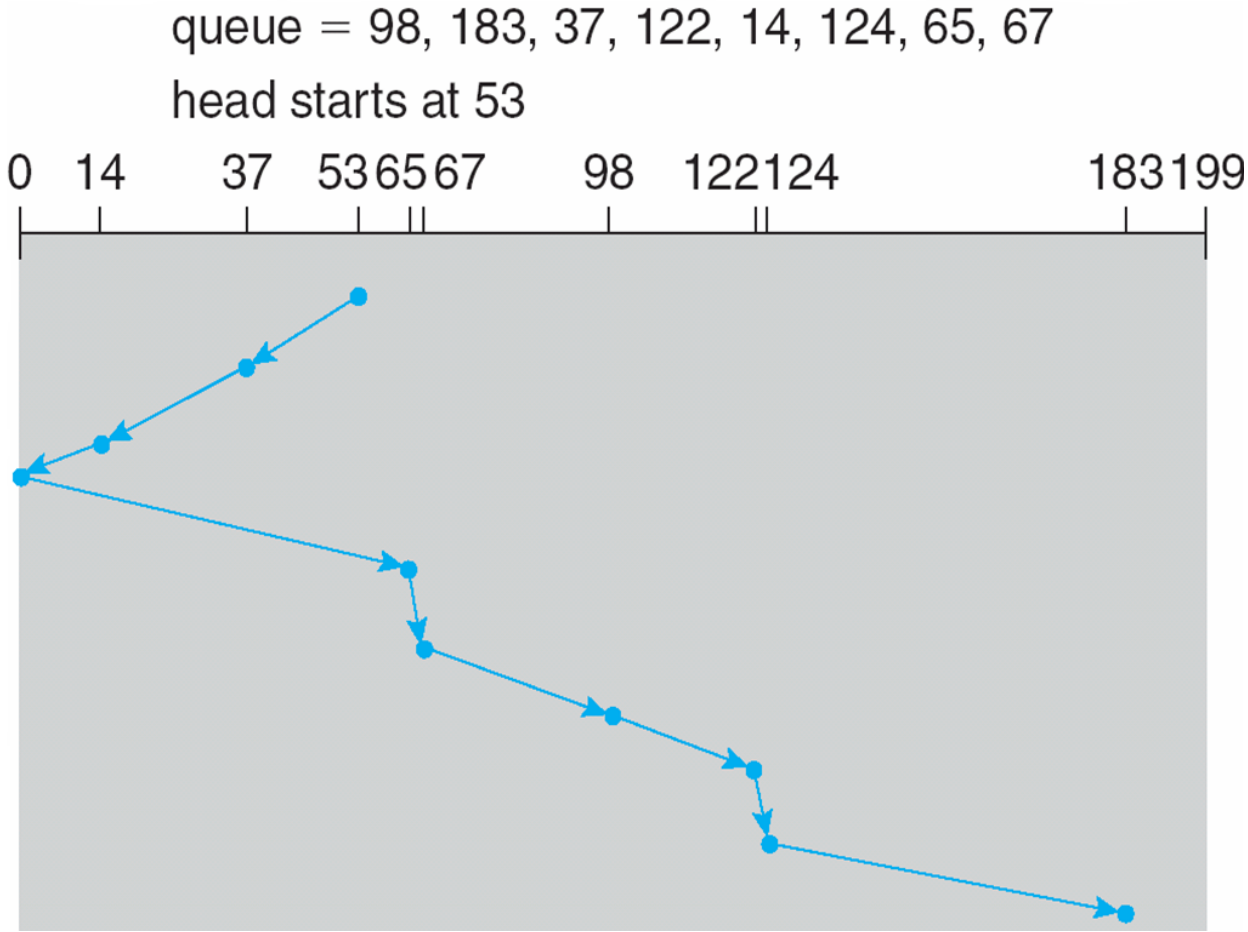


Algoritmi de planificare a accesului la disc

- SCAN
  - pentru acest algoritm, **mişcarea capetelor începe de la ultimul cilindru, mergând spre primul şi servind toate cererile pe care le întâlneşte.**
  - După ce a servit cererea de la cea mai mică adresă de cilindru solicitat, capetele îşi schimbă sensul de mers, servind cererile apărute ulterior până la servirea cererii de pe cilindrul solicitat cu cea mai mare adresă, apoi, iar se schimbă sensul ş.a.m.d.
  - Pentru exemplul nostru, dacă servirea cilindrului 53 s-a făcut la mişcarea spre adrese mici, atunci servirea se face în ordinea: 37, 14, 65, 67, 98, 122, 124, 183. Numărul de deplasări totale va fi de 192 cilindri.
  - Este posibil ca unele cereri să aştepte două parcurgeri ale discului până sunt servite.

Algoritmi de planificare a accesului la disc

- SCAN

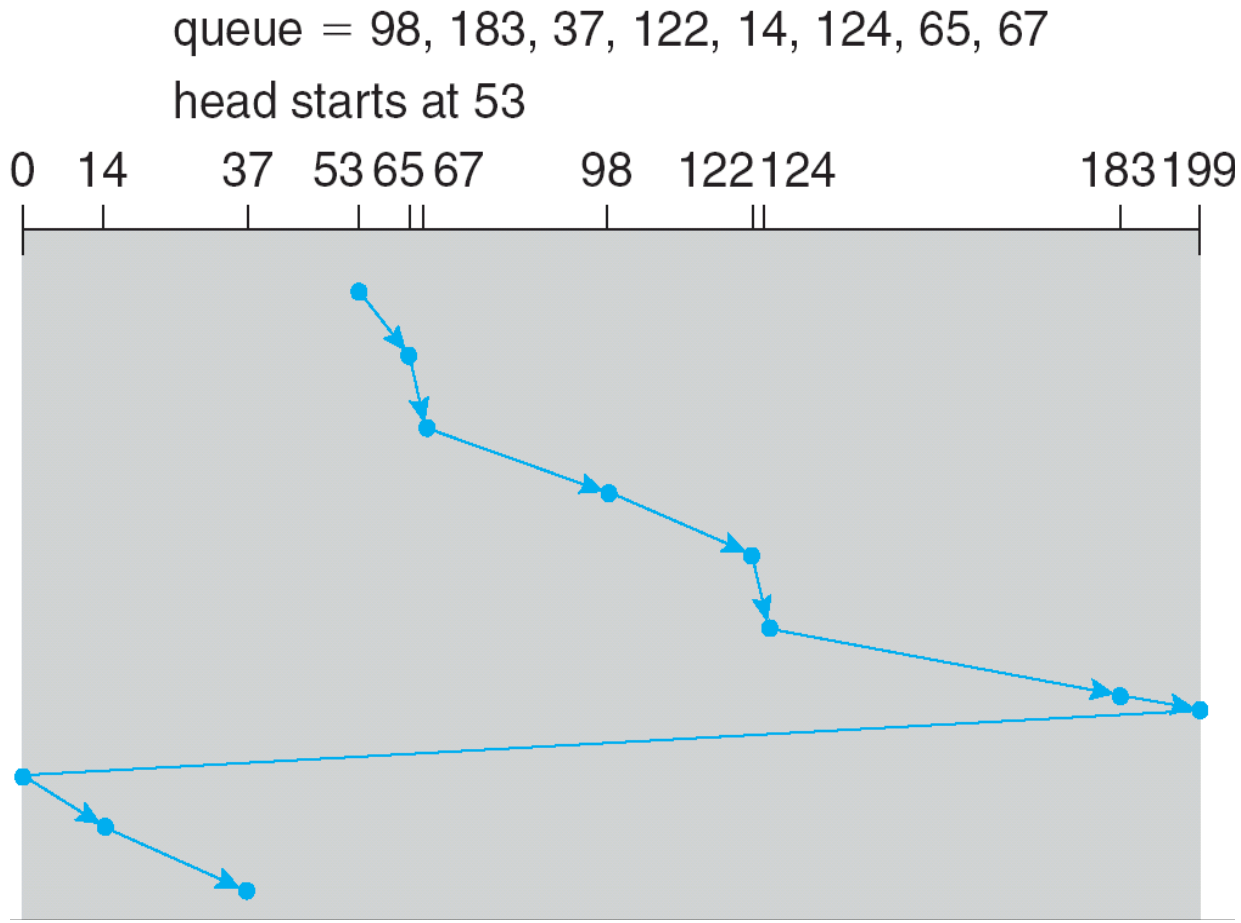


Algoritmi de planificare a accesului la disc

- C-SCAN (circular SCAN)
  - spre deosebire de algoritmul SCAN, **servirea cererilor se face numai când capetele se deplasează de la adrese mici spre adrese mari.**
  - Când se ajunge la ultimul cilindru, capetele sunt retrase automat pe cilindrul 0 (de regulă cu mare viteză), după care se reîncep servirile apărute în coadă.
  - Pentru exemplul nostru, ordinea de servire a cererilor este: 65, 67, 98, 122, 124, 183; 199; retragere la 0; 14, 37.
  - La unele tipuri de unităţi de disc retragerea capetelor pe cilindrul 0 durează mai puţin decât o poziţionare obişnuită şi în acest caz putem considera că fiecare cerere este servită cel mult după parcurgerea în întregime a cilindrilor discului.

Algoritmi de planificare a accesului la disc

- C-SCAN (circular SCAN)

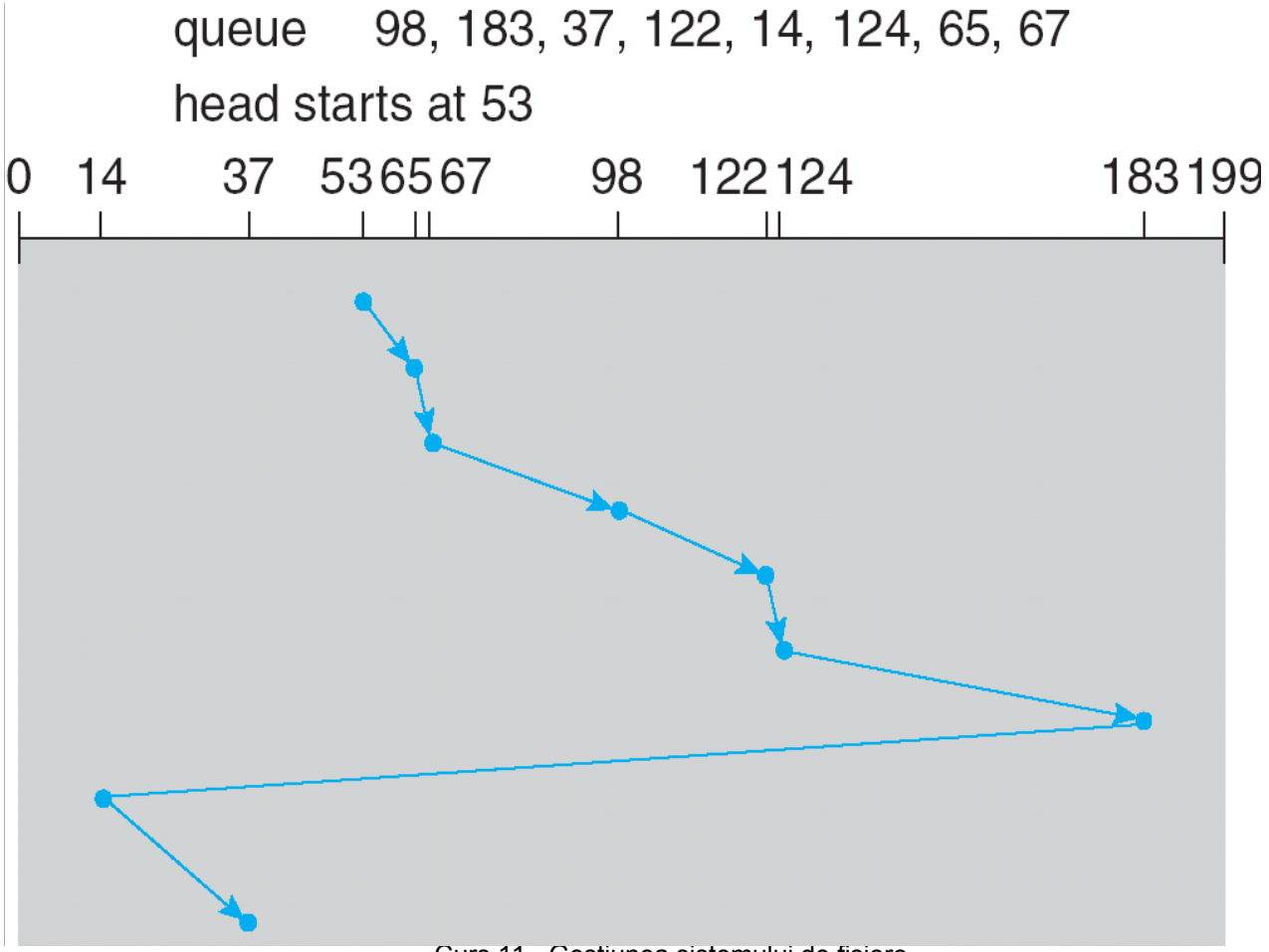


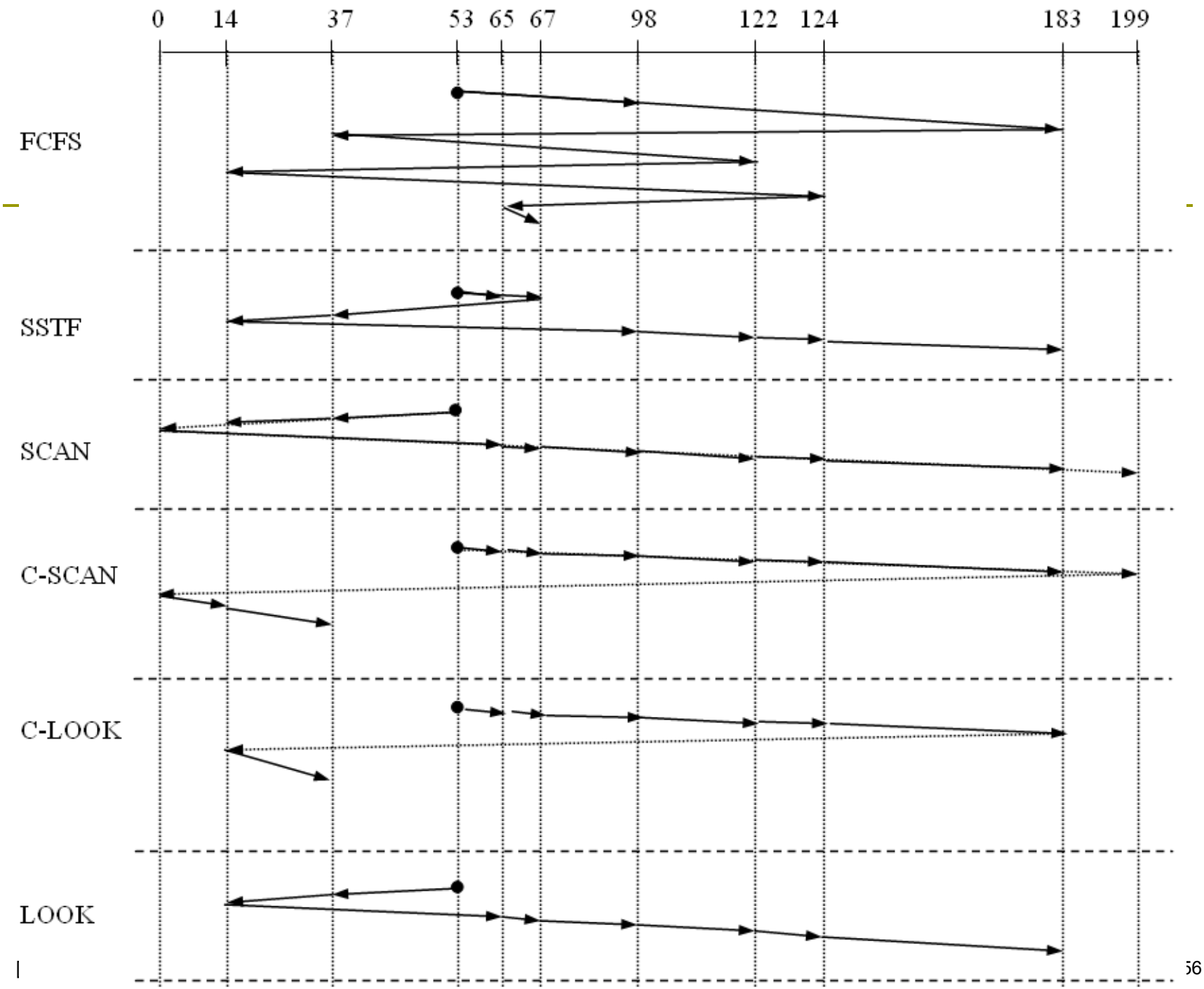
Algoritmi de planificare a accesului la disc

- LOOK
  - **capul de citire/scriere se deplasează spre cilindrul cel mai apropiat de acela la care se găseşte**
  - Exemplu: 37, 14, 65, 67, 98, 122, 124, 183
- C-LOOK
  - **capetele se deplasează cât de mult posibil în direcţia în care există cereri după care, dacă nu mai sunt cereri în direcţia curentă este schimbată direcţia.**
  - La schimbarea direcţiei nu este servită nici o cerere.
  - Exemplu: 65, 67, 98, 122, 124, 183, 14, 37

Algoritmi de planificare a accesului la disc

- C-LOOK




0
14
37
53
65
67
98
122
124
183
199
FCFS
SSTF
SCAN
C-SCAN
C-LOOK
LOOK

Algoritmi de planificare a accesului la disc

- Cererile de acces la disc sunt influenţate şi de metoda de alocare a fişierelor.
- localizarea directoarelor şi a blocurilor de index este foarte importantă, deoarece deschiderea unui fişier înseamnă parcurgerea structurii de directoare şi din acest motiv, plasarea directoarelor la jumătatea distanţei dintre marginile discului poate reduce semnificativ numărul de deplasări ale capetelor.
- **producătorii de discuri implementează algoritmii de servire a cererilor la disc în controller-ul de disc.**
- **Sistemul de operare va trimite cererile în ordinea FCFS, iar controller-ul le va prelua şi le va servi într-o ordine optimă.**

Managementul swap-ului

- implementarea spaţiului de swap trebuie realizată astfel încât să realizeze cele mai bune performanţe posibile.
- spaţiul de swap este utilizat în funcţie de felul în care sistemul de operare implementează algoritmii de acces la memorie.
  - De exemplu, poate fi folosit pentru păstrarea întregii imagini a procesului inclusiv zonele de cod şi date.
- Sistemele de operare pot oferi posibilitatea utilizării mai multor spaţii de swap.

Managementul swap-ului

- Spaţiul de swap poate fi localizat în sistemul normal de fişiere ca un simplu fişier de dimensiune mare (cazul Windows) şi, în acest caz, rutine de acces la fişier pot fi folosite pentru utilizarea swap-ului.
- adăugarea unui nou spaţiu de swap se poate face prin crearea unui nou fişier.
- această implementare nu este foarte eficientă datorită faptului că navigarea prin structura de directoare a sistemului consumă mult timp şi necesită accese suplimentare la disc.
- fragmentarea externă poate duce la creşterea timpului de citire şi scriere din swap.

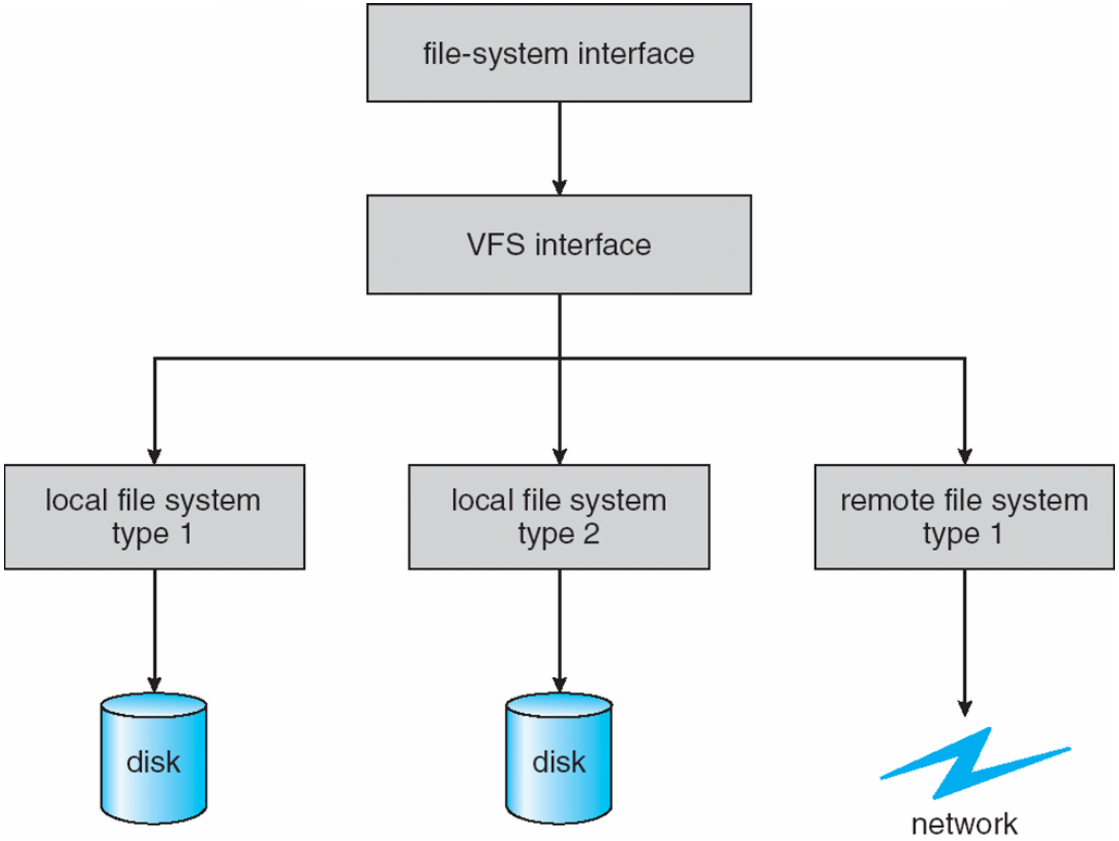
Managementul swap-ului

- O altă metodă este crearea unei partiţii speciale pentru swap.
- Avantajul aceste metode este viteza.
- Algoritmii sunt optimizaţi pentru o extragere şi o stocare foarte rapidă a datelor.
- Fragmentarea internă este destul de mare, dar acest lucru este acceptabil, deoarece timpul de viaţă al datelor în swap este destul de redus decât într-un sistem de fişiere.
- Este permisă şi existenţa unor fişiere suplimentare de swap.

Virtual File Systems

- Reprezita o modalitate de implementarea a sistemelor de fisiere in diverse sisteme de operare, asemanatoare principiilor POO
- Permit utilizarea acelorasi apeluri sistem pentru pentru a accesa diverse sisteme de fisiere
- apelurile sistem se fac catre VFS si nu catre un anume sistem de fisiere

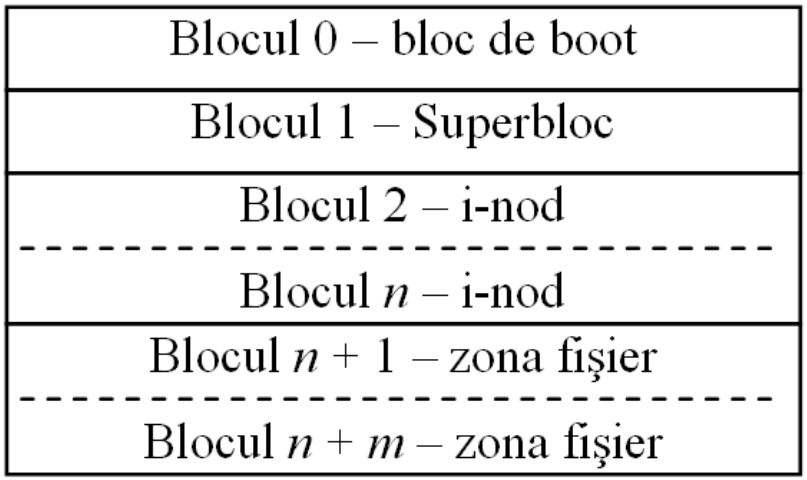
Virtual File Systems

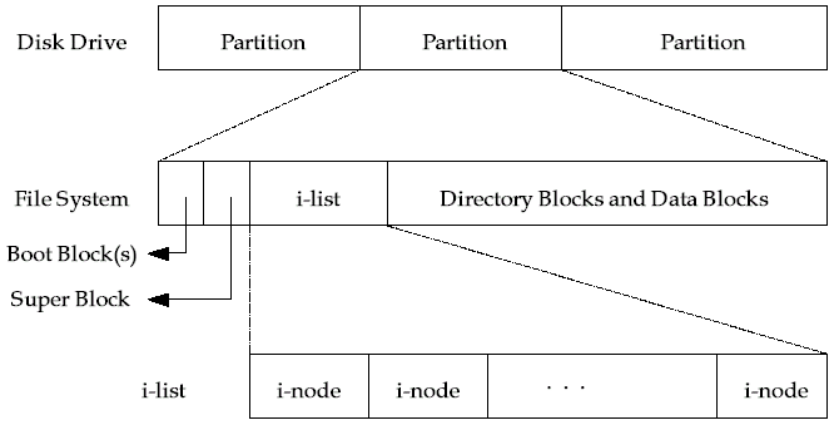


Studii de caz - Unix

- Sistemul de fişiere Unix este o structură de date rezidentă pe disc. Şi este formată din 4 categorii de blocuri:
  - Blocul 0 – conţine programul de încărcare al SO;
  - Blocul 1 sau superbocul conţine o serie de informaţii prin care se defineşte sistemul de fişiere de pe disc: numărul de i-noduri, numărul de zone definite pe disc, pointeri spre harta de biţi a alocării i-nodurilor, pointeri spre harta de biţi a spaţiului liber disc, dimensiunile zonelor disc, etc. ;
  - Blocurile de la 2 la n (n - este o constantă a formatării discului) conţin lista de i-noduri (index-node), listă numită şi i-listă.
  - i-nodul este o structură de date ce reprezintă, de fapt, descriptorul de fişier. Numărul de ordine al unui i-nod în cadrul i-listei se numeşte i-număr.

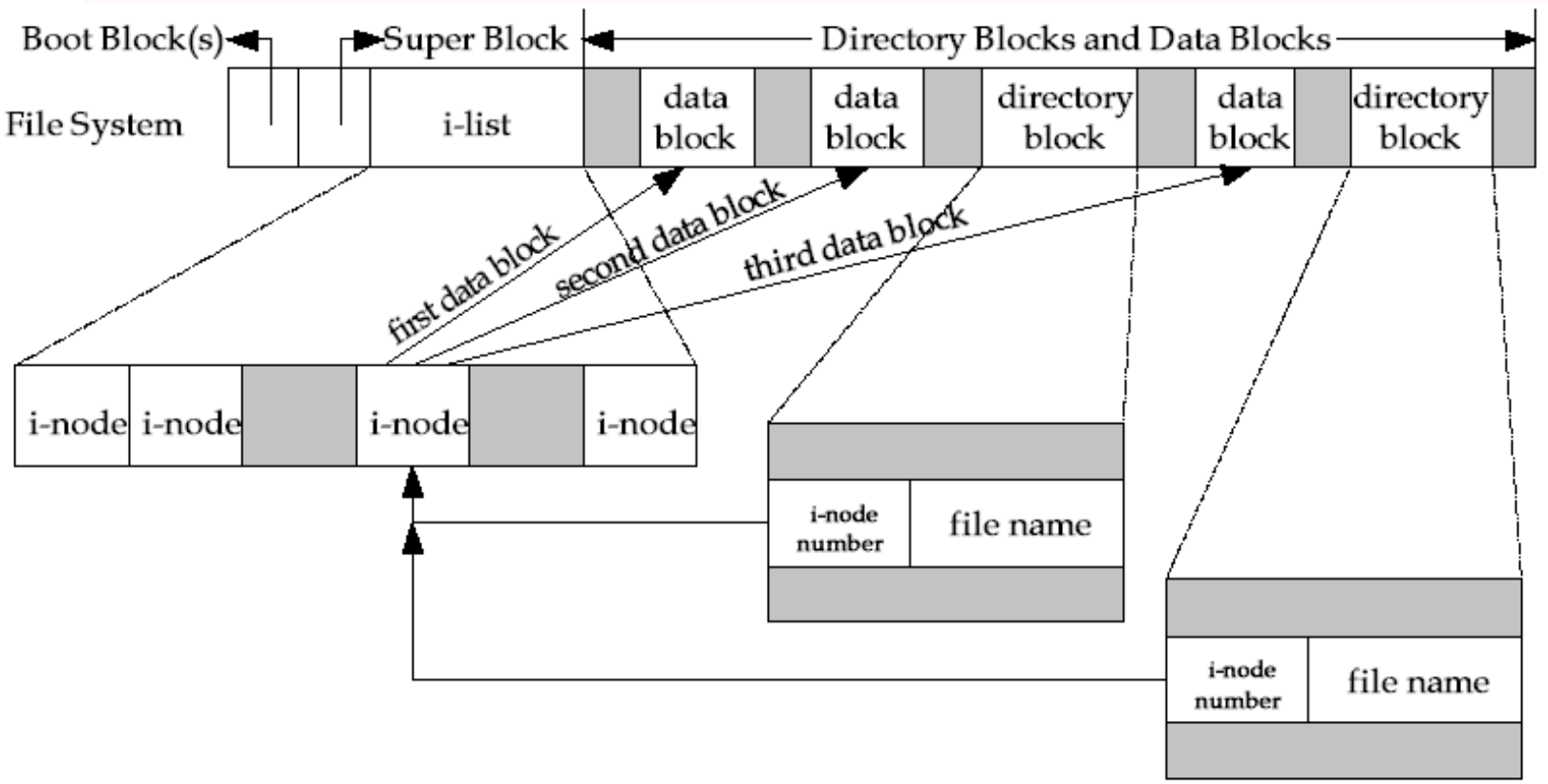
Studii de caz - Unix





Studii de caz - Unix

- Structura detaliată a unui disc



Studii de caz - Unix

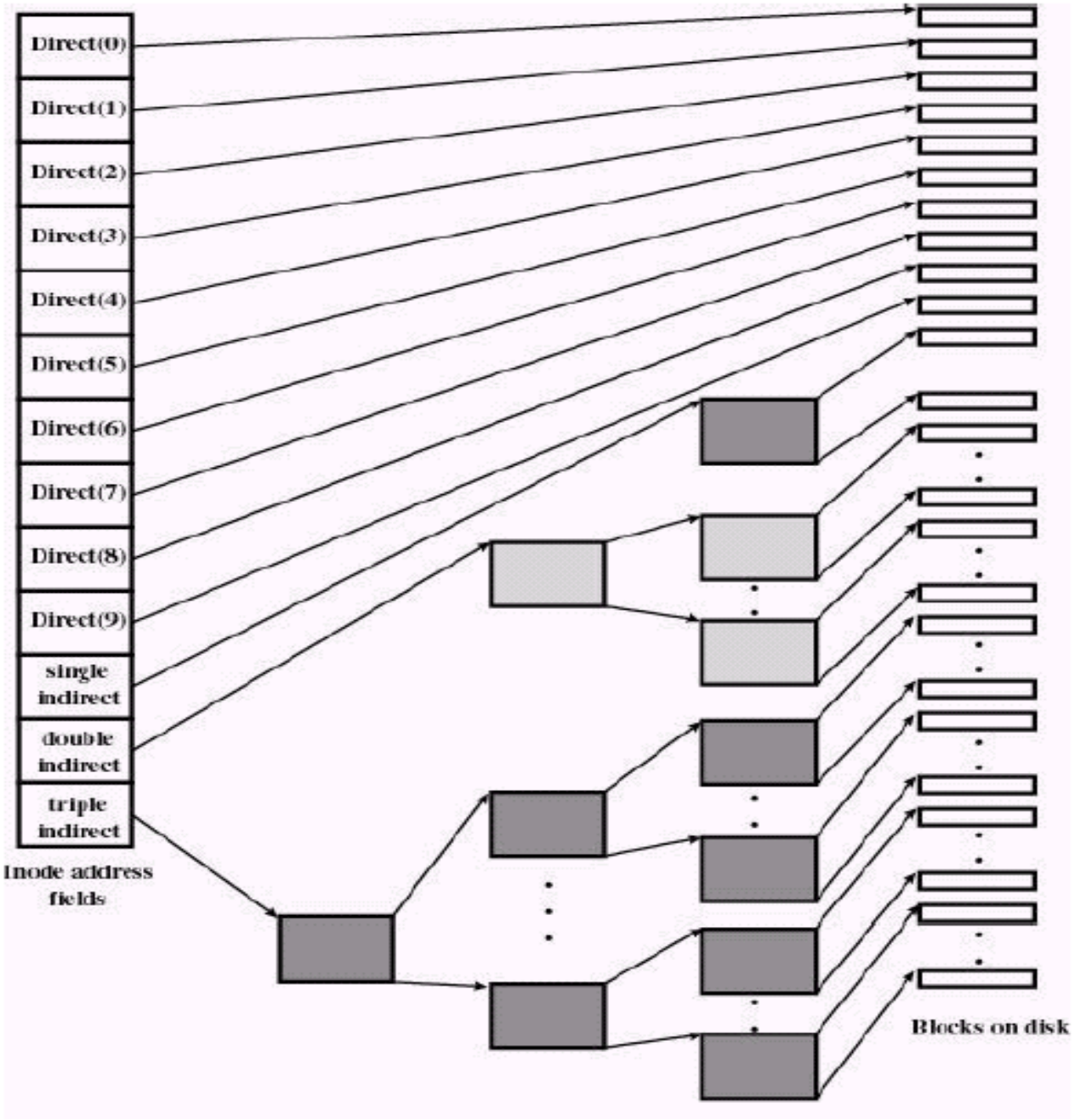
- Structura detaliată a unui disc
  - Partea cea mai mare a discului este rezervată zonei fişierelor.
  - În superbloc este realizată evidenţa spaţiului liber şi acest lucru se face prin metoda fişierului invers.
  - Informaţiile necesare alocării sunt fixate în i-noduri.

Studii de caz - Unix

- Structura detaliată a unui disc
  - Structura unui director este de forma:

Numele fişierului	i-număr

  - Informaţiile care apar într-un i-nod sunt:
    - identificatorul proprietarului fişierului,
    - identificatorul de grup,
    - biţii de protecţie,
    - lungimea fişierului,
    - data creării şi a ultimei actualizări,
    - numărul de legături la fişiere,
    - indicarea faptului că fişierul este un director,
    - pointerii spre blocurile alocate fişierului (13 intrări)


Direct(0)
Direct(1)
Direct(2)
Direct(3)
Direct(4)
Direct(5)
Direct(6)
Direct(7)
Direct(8)
Direct(9)
single indirect
double indirect
triple indirect
Inode address fields
Blocks on disk

Studii de caz – Unix

- Structura unui i-nod
  - Primele 10 intrări
    - adresele primelor 10 blocuri de câte 512 octeţi, care aparţin fişierului
  - Intrarea 11:
    - un bloc de indirectare simplă, ce conţine adresele următoarelor 128 blocuri de câte 512 octeţi care aparţin fişierului
  - Intrarea 12:
    - adresa unui bloc de indirectare dublă, care conţine adresele a 128 blocuride indirectare simplă,
    - fiecare bloc conţine adresele a câte 128 blocuri de indirectare de câte 512 octeţi, fiecare cu informaţii ce aparţin fişierului
  - Intrarea 13
    - conţine adresa unui bloc de indirectare triplă, care conţine adresele a 128 blocuri de indirectare dublă.
    - Numărul de accese necesare pentru a obţine direct un octet oarecare este cel mult 4, iar pentru fişiere mici acest număr este mai mic.

Studii de caz - Unix

- Alocarea fişierelor: bazată pe blocuri, este o alocare dinamică.
  - Este utilizat un index pentru a şti în orice moment localizarea pe disc a părţilor din fişier.
  - Lungimea unui bloc pentru UNIX System V este de 1Kbyte
  - fiecare bloc poate stoca 256 adrese de blocuri
  - dimensiune maximă posibilă a unui fişier de 16Gbytes.
- Avantaje:
  - i-nodul are o dimensiune fixă, relativ mică şi, din acest motiv, poate fi păstrat în memorie un timp mai îndelungat;
  - fişierele mici pot fi accesate cu indirectare simplă sau fără indirectare, lucru ce duce la reducerea timpului de acces la fişier;
  - Dimensiunea maximă a unui fişier este suficient de mare pentru a satisface orice aplicaţie.

Studii de caz – JFS (Journal File System)

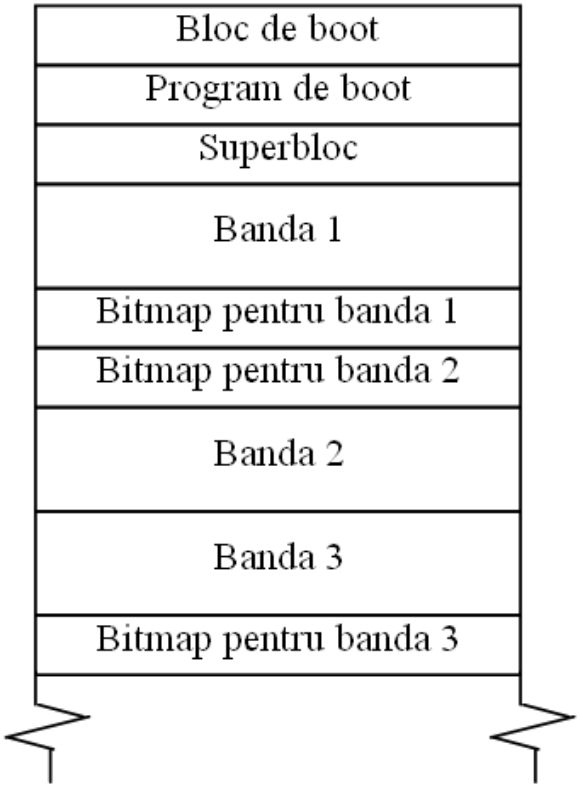
- Dezvoltarea sistemelor de calcul şi a utilizării lor fără întrerupere a dus la necesitatea apariţiei unei metode de protecţie a sistemului de fişiere în caz de accidente.
- au apărut tehnici de păstrare a informaţiilor legate de toate operaţiunile realizate cu discul (**jurnale** sau **log**-uri)
- Astfel, **scrierile pe disc sunt realizate asincron** datorită faptului că, după terminarea unei cereri de scriere la disc, **informaţiile sunt înregistrate în log-uri şi apoi făcute modificările în tabelele descriptorilor de fişier**.

Studii de caz – JFS (Journal File System)

- Informaţiile din log-uri sunt actualizate astfel:
  - este folosit un cursor (pointer)
  - Atunci când cursorul înaintează odată cu scrierea pe disc a blocurilor asociate tranzacţiei şi informaţiile legate de tranzacţia respectivă sunt şterse din log-uri
  - Pentru refacerea informaţiei se reiau toţi paşii de la ultima poziţie a cursorului
- Avantaje:
  - refacerea foarte rapidă a datelor – depinde de dimensiunea log-urilor şi nu de dimensiunea sistemului de fişiere.
- Dezavantaje:
  - scrieri suplimentare pe disc;
  - pierderea de spaţiu liber pentru păstrarea log-urilor (nesemnificativ în raport cu posibilităţile de refacere a informaţiei)

Studii de caz – HPFS
High Performance File System

- Structura de date folosită pentru descrierea localizării pe disc a fişierelor este B-arborele
- organizează discul în volume şi rezervă primele 18 sectoare pentru blocul de boot, superbloc şi blocul "de rezervă".
- Aceste blocuri conţin informaţiile de control care sunt utilizate pentru iniţializarea sistemului, gestiunea sistemului de fişiere şi refacerea sistemului de fişiere după producerea de erori.



Studii de caz – HPFS
High Performance File System

- Restul sistemului de fişiere este organizat în benzi de 8MB, fiecare având asociată o hartă de biţi de dimensiune 2KB în care fiecare bit corespunde unui bloc de 4 KB din cadrul benzii şi indică prin 0 sau 1 faptul că blocul respectiv este liber sau alocat.
- Hărţile de biţi sunt situate alternativ la sfârşitul şi la începutul benzilor aşa încât să poată fi făcute alocări contigue de până la 16 MB (două benzi consecutive).

Studii de caz – HPFS
High Performance File System

- Pentru descrierea fişierelor, HPFS utilizează F-noduri.
- Acestea sunt structuri de date care conţin atributele şi informaţiile legate de localizarea pe disc a fişierelor.
- Fiecare F-nod conţine referiri la cel mult 8 extensii.
- Extensiile sunt entităţi distincte stocate în fiecare volum, fiecare din ele putând adresa blocuri ce însumează până la 16 MB.
- pot fi suportate dimensiuni ale fişierelor de până la 128MB.
- Pentru fişierele mai mari, F-nodurile conţin un număr de 12 adrese către noduri de alocare, ce pot fi utilizate pentru a adresa mai multe extensii.

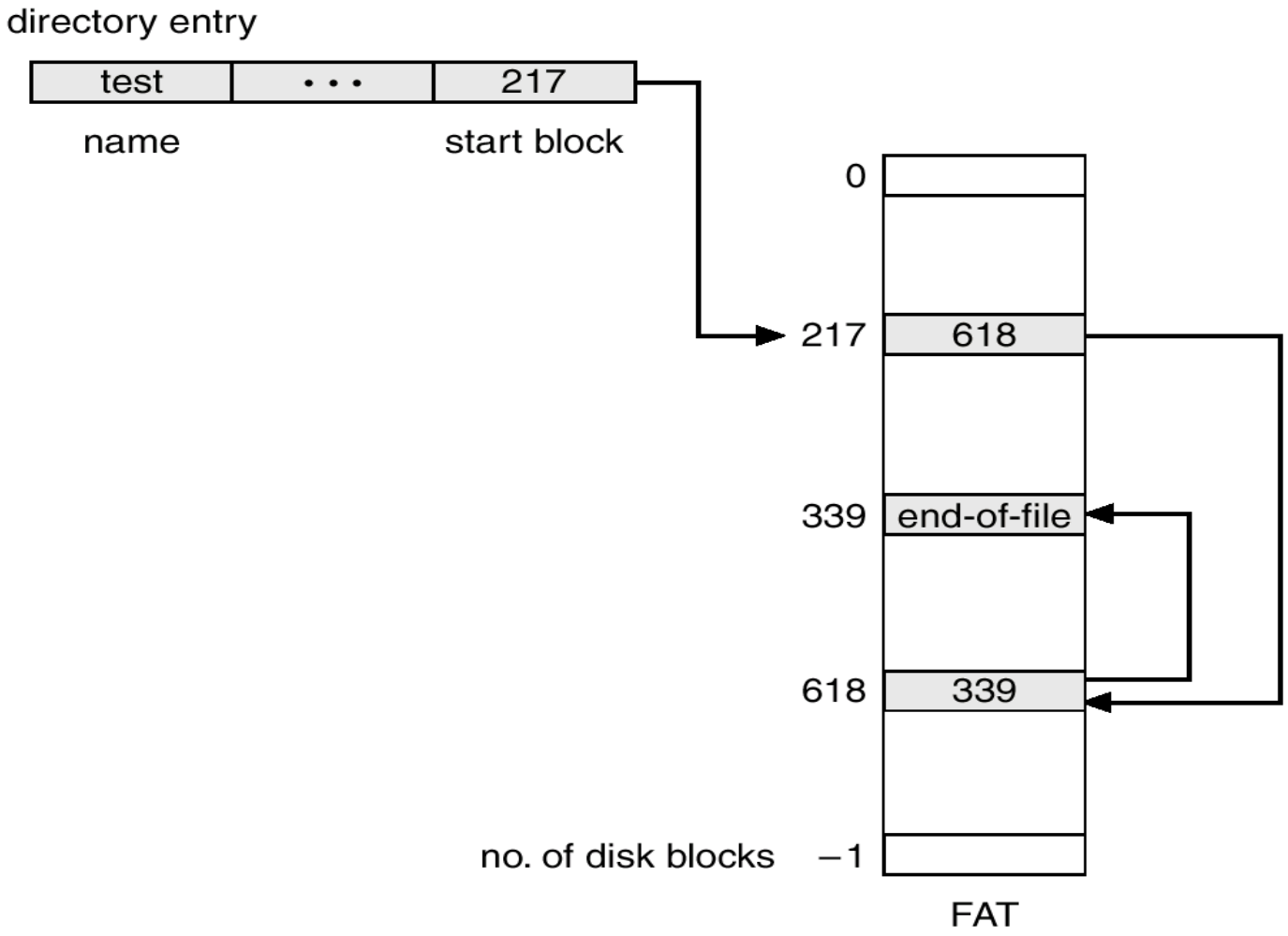
Studii de caz – Windows

- Windows NT oferă posibilitatea optării pentru 2 sisteme de fişiere diferite:
  - NTFS
  - FAT16.
- Windows 2000/XP si ulterioare
  - NTFS – imbunatatit fata de versiunea de la Windows NT
  - FAT32

Studii de caz – Windows

- File Allocation Table (FAT)
  - Sistemul de fişiere FAT foloseşte ca mecanism de alocare lista înlănţuită de indecşi.
  - Sistemul de operare întreţine o tabelă de alocare ce conţine câte o intrare pentru fiecare bloc memorat pe disc.
  - Tabela FAT are rolul de a reda modul de înlănţuire a blocurilor ce alcătuiesc un anumit fişier, memorând în fiecare intrare din FAT indexul blocului următor.

Studii de caz – Windows File Allocation Table (FAT)



Studii de caz – Windows

- File Allocation Table (FAT)
  - O alternativă a acestei metode de alocare memorează pointerii către blocul următor chiar în blocurile de date, renunţând la utilizarea unei liste separate.
  - Avantajul acestei reprezentări mai compacte conduce la alte două dezavantaje majore:
    - Primul este legat de accesul aleator la datele conţinute în fişier:
      - înlănţuirea ar trebui urmărită citind practic bloc-cu-bloc, adică efectuând un număr mare de operaţii de I/O, în timp ce păstrarea tabelei FAT în memorie conduce la obţinerea unei viteze acceptabile.
    - Al doilea dezavantaj se referă la dimensiunea datelor dintr-un bloc, care nu mai este multiplu de doi ci diferenţa dintre dimensiunea unui bloc – multiplu de doi - şi un număr de octeţi alocaţi pointerului către blocul următor.
  - Deşi pare o problemă minoră, totuşi în multe situaţii acest fapt poate conduce la penalizări asupra eficienţei operaţilor de I/O (care vizează în general transferul unor structuri de date de dimensiune putere a lui doi).

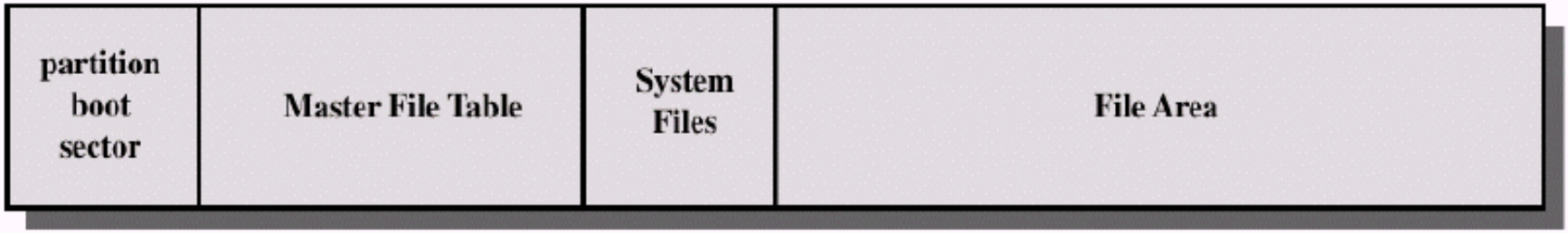
Studii de caz – Windows

- New Tehnology File System (NTFS)
  - Sistemul de fişiere NTFS foloseşte pentru descrierea localizării pe disc a fişierelor o structură de tip B-arbore.
  - Fiecare volum NTFS conţine o **Master File Table** (**MFT**) care este, de fapt, un fişier ce cuprinde informaţii despre fişierele şi directoarele volumului respectiv.
  - **MFT** este organizată ca o succesiune de înregistrări dintre care primele 16 sunt utilizate pentru descrierea **MFT** însăşi şi pentru furnizarea de informaţii necesare refacerii după situaţii de avarie.
  - Următoarele înregistrări din **MFT** descriu fişierele şi directoarele.
  - Dacă au o dimensiune suficient de mică (cel mult 1500 octeţi), informaţiile despre fişiere şi directoare sunt înglobate într-o singură înregistrare **MFT**, iar în caz contrar înregistrările **MFT** vor conţine adrese către una sau mai multe extensii.

Studii de caz – Windows

- New Tehnology File System (NTFS)

Structura unui volum NTFS



Studii de caz – Windows

- New Tehnology File System (NTFS)
  - NTFS asigură suport pentru spaţiul de nume şi pentru dimensiuni mari ale fişierelor şi volumelor,
  - asigură un mecanism de securitate comparabil cu al Unix-ului prin intermediul listei de control a accesului (access control list – ACL) asupra fişierelor şi a directoarelor la nivel de utilizator.
  - Se asociază drepturi specifice utilizatorilor şi grupurilor asupra acţiunilor pe care utilizatorii le pot efectua asupra fişierelor şi directoarelor, permiţând distribuţia drepturilor fără a da acces la tot sistemul.

Studii de caz

- Sisteme de fişiere distribuite
  - permit distribuirea sistemului de fişiere pe maşini diferite fizic, păstrând totuşi datele disponibile de pe aceste maşini.
  - Un avantaj imediat este backup-ul uşor al datelor şi managementul acestora.

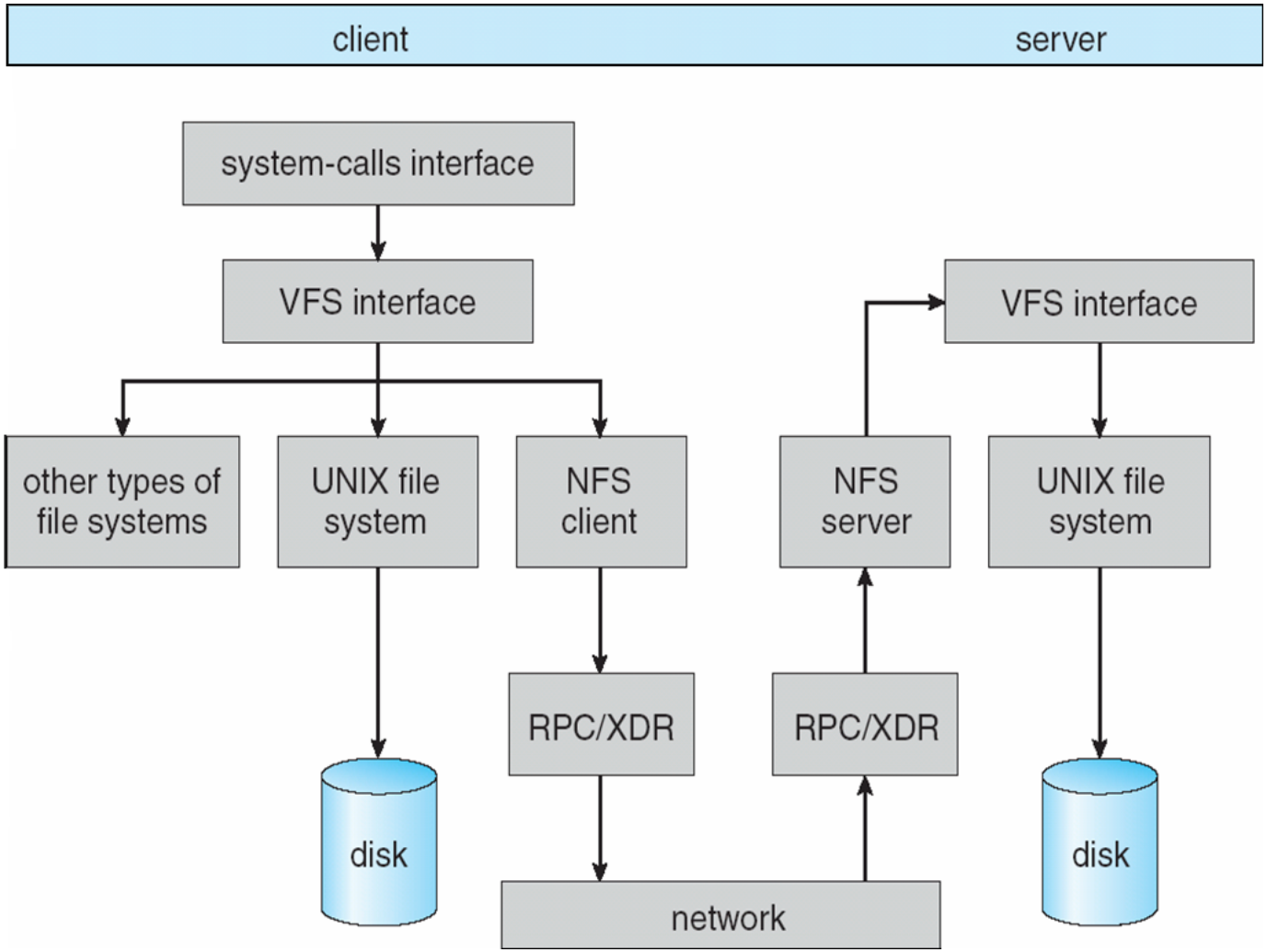
Studii de caz

- Sisteme de fişiere distribuite
  - NFS (Network File System dezvoltat de SUN Microsystems)
  - RFS (dezvoltat de AT&T) pe platforme UNIX
  - Microsoft DFS: NTFS si Active Directory.
  - Novel NetWare File System
  - AFS – AndrewFileSystem (Carnegie Mellon University si IBM)
  - Open AFS – implementare open source a AFS
  - CODA – derivat din AFS2, dezvoltat la Carnegie Mellon University
  - Hadoop si CloudStore (implementari open source in Java/C++ a Google File System)
  - Amazon S3
  - Lustre –open source; dezvoltat de Cluster File Systems si preluat de SUN Microsystem
  - PVFS – open source; devoltat de The Parallel Architecture Research Laboratory at Clemson University, Mathematics and Computer Science Division at Argonne National Laboratory, si Ohio Supercomputer Center.
  - Global File System (GFS si GFS2) - RedHat
  - General Parallel File System – IBM; suport pentru AIX,Linux, Windows

Studii de caz - NFS

- Sisteme de fişiere distribuite – NFS
  - Foloseste RPC (remote procedure call)
  - Pentru functionare trebuie sa fie active serviciile:
    - portmap,
    - rpcgssd, rpcidmapd, rpcsvcgssd
    - nfs, nfslock
  - NFS v3:
    - nu menţine starea fişierelor deschise pe un server;
    - este mai puţin eficient, deoarece serverul de fişiere trebuie să redeschidă fişierele pentru fiecare tranzacţie realizată prin reţea, lucru ce poate fi îmbunătăţit prin folosirea cache-ului;
    - Refacerea după o cădere a unui server este transparentă clientului, deoarece serverul nu are nici o stare de refăcut.

Studii de caz - NFS

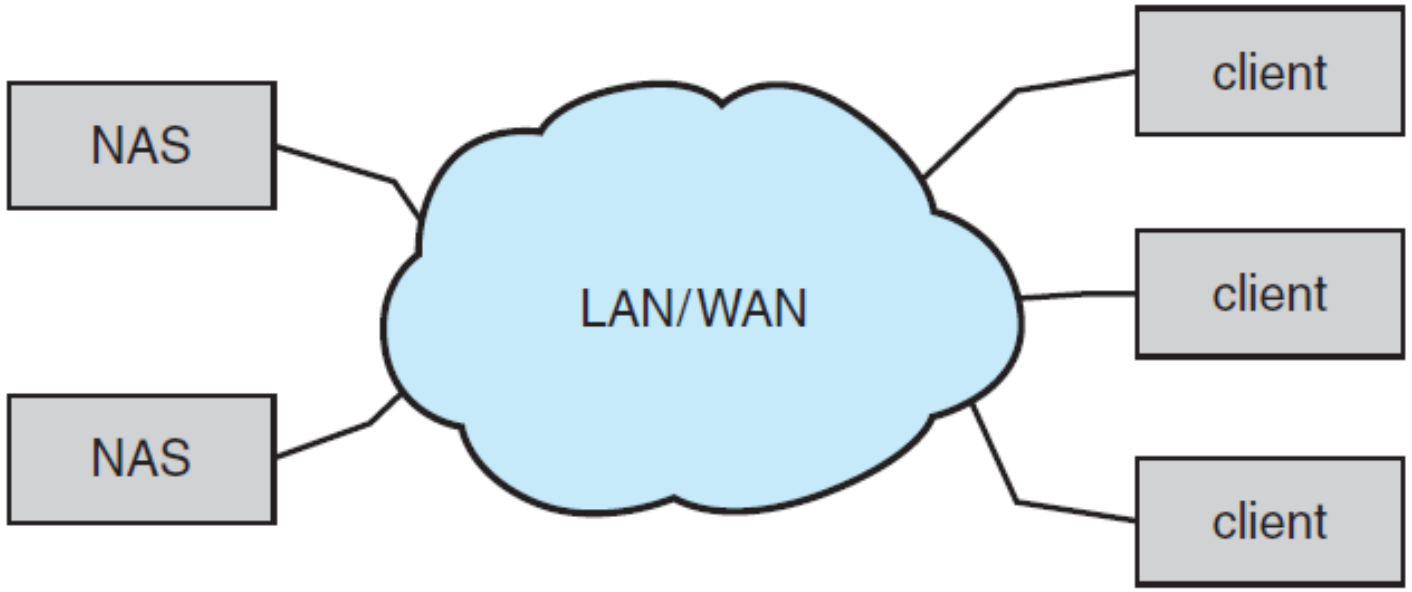


Dispozitive de stocare

- Sistemele de calcul acceseaza dispozitivele de stocare in doua moduri
  - Prin porturile de I/O (host-attached storage);
  - Prin conexiuni în reţea, adeseori referite ca network-attached storage

Network-Attached Storage (NAS)

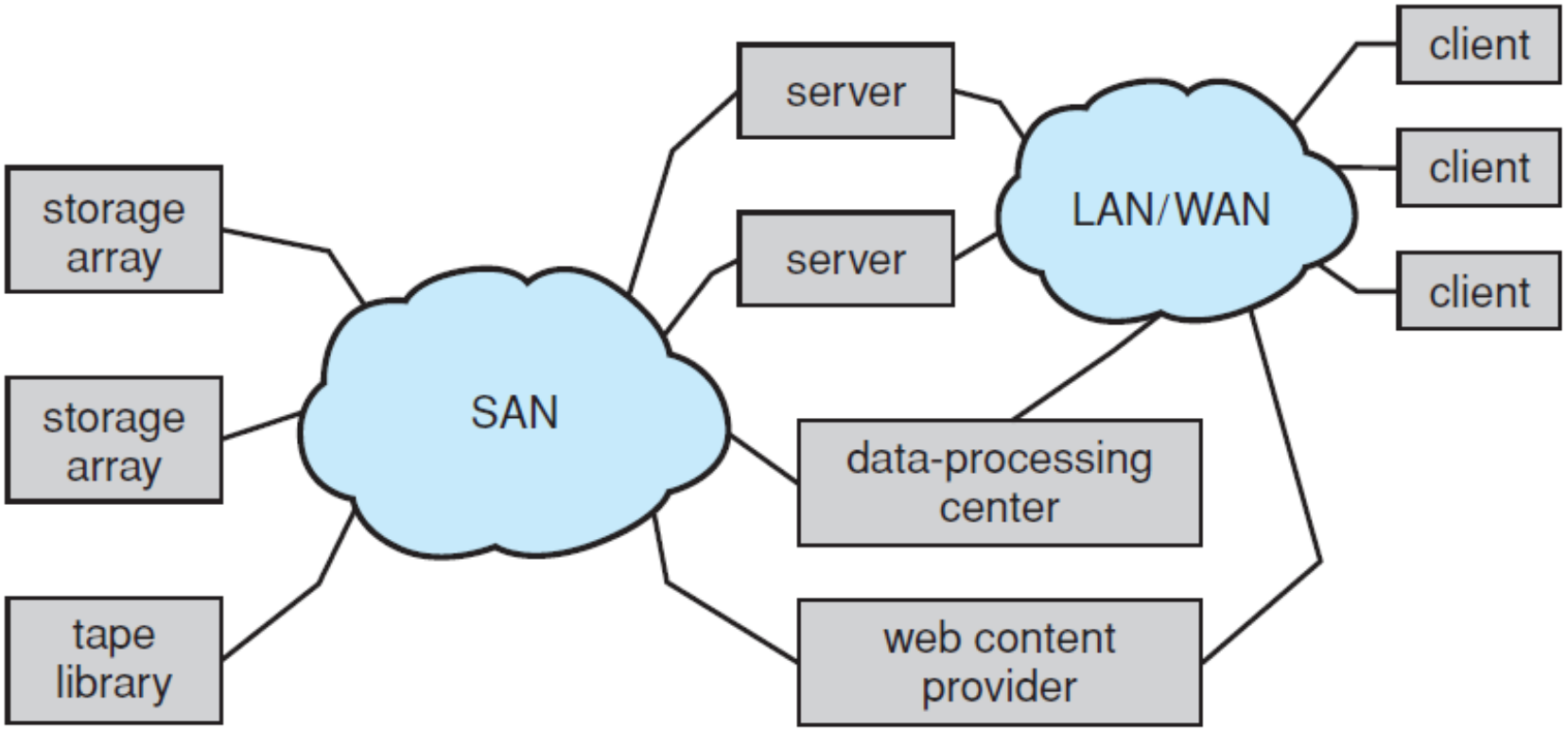
- Sistem de stocare specializat accesat peste interfete de comunicaţii de date



Storage-Area Network

- storage-area network (SAN) – **reţea de date privată ce foloseşte protocoale de comunicaţii pentru dispozitive de stocare** in locul protocoalelor de comunicaţie în reţea, ce conecteaza servere şi unitaţi de stocare

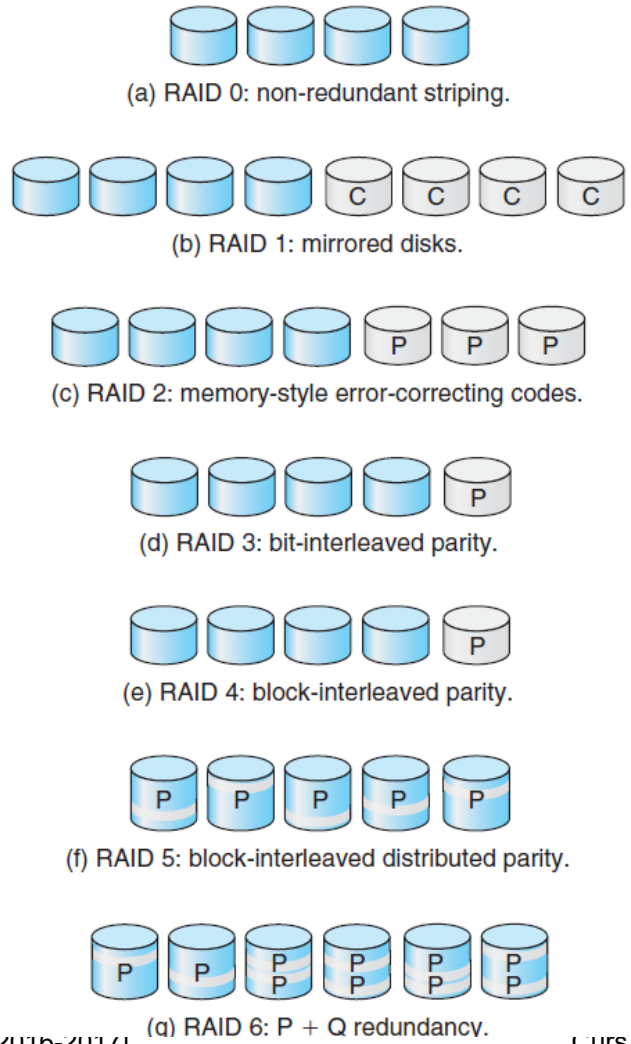
Storage-Area Network



Disponibilitatea datelor

- Pentru asigurarea disponibilităţii datelor sunt folosite diferite tehnici de organizare a discurilor numite **redundant arrays of independent disks (RAID)**

Configuratii RAID



(a) RAID 0: non-redundant striping.

(b) RAID 1: mirrored disks.

(c) RAID 2: memory-style error-correcting codes.

(d) RAID 3: bit-interleaved parity.

(e) RAID 4: block-interleaved parity.

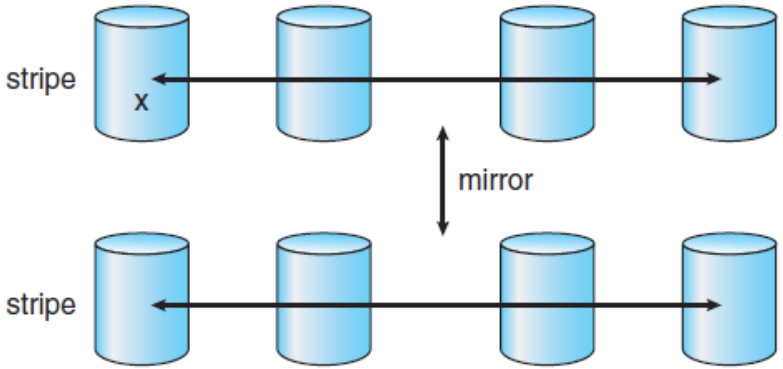
(f) RAID 5: block-interleaved distributed parity.

(g) RAID 6: P + Q redundancy.

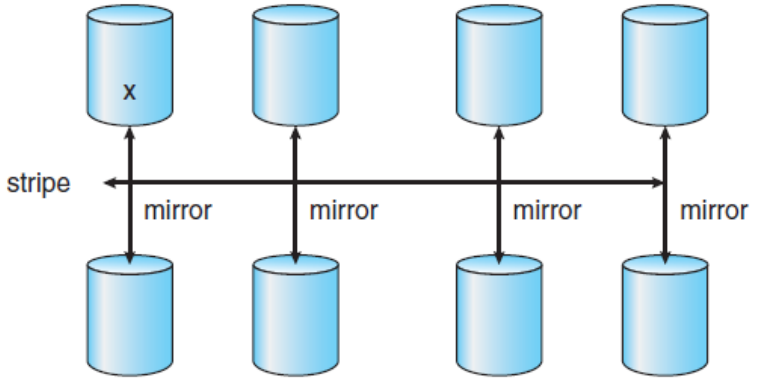
- P - error-correcting bits
- C - a second copy of the data

Sursa: Silberschatz A., Galvin P. - Operating System Concepts, 9th Edition , John Wiley & Sons, 2012

Configuratii RAID



- RAID 0+1



- RAID 1+0

Referințe

- http://en.wikipedia.org/wiki/Comparison_of_file_systems
- http://en.wikipedia.org/wiki/List_of_file_systems
- http://en.wikipedia.org/wiki/List_of_file_systems#Distributed_file_systems