Sisteme de gestiune a fisierelor

**Cuprins:**

**1. Principii ale sistemelor de gestiune de fisiere**

(Stanescu Robert)

**2. Gestiunea spatiului de memorie externa**

(Banciu Alexandru)

**3. Sisteme de gestiune de fisiere Windows (FAT, NTFS, etc.)**

(Pavaloaia Oriana)

**4.** . **Sisteme de gestiune de fisiere UNIX**

(Sasu Alexandru)

**5. Sisteme de gestiune de fisiere in alte sisteme de operare**

(Chirila Vlad)

**1. Principii ale sistemelor de gestiune de fisiere**

**(Stanescu Robert)**

**1.1. Introducere**

Un sistem de gestiune fisiere (filesystem) este o metoda de stocare si organizare a fisierelor electronice si a datelor lor. In esenta, el organizeaza aceste fisiere intr-o baza de date pentru stocare, organizare, manipulare, si recuperare de catre sistemul de operare al computerului.

Sistemele de gestiune de fisiere sunt folosite la dispozitive de stocare de date cum ar fi un hard disk-uri sau CD-ROM-uri pentru a mentine locatia fizica a fisierelor. Mai mult, ele ar putea oferi acces la datele de pe un server de fisiere, actionand in calitate de clienti ai unui protocol de retea (ex: NFS , SMB sau 9p), sau pot fi virtuale existand numai ca o metoda de acces pentru date virtuale (ex: procfs ). A se face diferenta fata de un serviciu de directoare sau de registrii.

**1.2. Aspecte ale sistemelor de gestiune** **de fisiere**

Cele mai multe sisteme de gestiune de fisiere utilizeaza un suport de stocare a datelor care ofera acces la o serie de sectoare de marime fizica fixa, in general putere a lui 2 (512 bytes sau 1, 2, 4 KB sunt cele mai frecvente). Sistemul de gestiune de fisiere este responsabil pentru organizarea acestor sectoare in fisiere si directoare , si tine evidenta care sectoare apartin carui fisier si care nu sunt utilizate. Majoritatea sistemelor de gestiune de fisiere adreseaza datele in unitati de dimensiuni fixe numite " grupuri "sau" blocuri ", care contin un anumit numar de sectoare de disc (de obicei intre 1 si 64). Aceasta este cea mai mica cantitate de spatiu pe disc, care poate fi alocata unui fisier.

Cu toate acestea, sistemele de gestiune de fisiere nu au neaparata nevoie de un dispozitiv de stocare. Un sistem de gestiune de fisiere pot fi folosit pentru a organiza si reprezenta acces la toate datele, indiferent daca aceastea sunt memorate sau generate dinamic (ex: procfs ).

**Nume de fisiere**

Un nume de fisier este un nume atribuit unui fisier pentru a-i asigura o locatie de stocare in memoria computerului. Prin acest nume un fisier poate fi accesat. Indiferent daca sistemul de gestiune de fisiere are un dispozitiv de stocare de baza sau nu, sistemele de gestiune de fisiere au in mod normal directoare care asociaza numele cu fisierul, de obicei prin introducerea numelui lui intr-un index al unui tabel de alocare a fisierelor de un anume fel, cum ar fi FAT pentru DOS, sau un *inode* pentru un sistem de gestiune de fisiere gen Unix. Structurile directorului pot fi plate, sau pot permite ierarhii, unde directoarele pot contine subdirectoare. In unele sisteme de gestiune de fisiere, numele fisierelor sunt structurate, cu o sintaxa speciala pentru extensii si versiuni. In altele, ele sunt siruri simple de caractere, iar metadatele fisierelor sunt stocate in alta parte.

**Metadate**

Alte atribute sunt de obicei asociate fiecarui fisier intr-un sistem de gestiune de fisiere. Lungimea datelor continute intr-un fisier poate fi stocata ca un numar de blocuri alocate fisierului sau ca un numar exact de bytes. Timpul cand fisierul a fost modificat ultima data poate fi stocat ca un *timestamp* (stampila de timp) al lui. Unele sisteme de gestiune de fisiere, de asemenea pot stoca timpul creerii fisierului, cand a fost ultima data accesat, sau cand metadatele fisierului au fost modificate. (Obs: Multe sisteme de operare mai vechi nu tineau cont de astfel de detalii legate de timp) Alte informatii pot include tipul de dispozitiv al fisierului (ex: bloc , caracter , socket , subdirector , etc), ID-ul ultilizatorului si ID-ul grupului , si setari legate de permisiunea de acces (ex: daca fisierul este read-only, executabil etc.)

Atribute arbitrare pot fi asociate in sisteme de gestiune de fisiere avansate, cum ar fi NTFS , XFS , ext2 / ext3 , unele versiuni ale UFS , si HFS + , folosind atribute de fisier prelungite . Aceasta facilitate este implementata in nucleele sistemelor de operare Linux , FreeBSD si Mac OS X, si permite metadatelor sa fie asociate cu fisierul la nivelul de sistem de gestiune de fisiere. Acest lucru, de exemplu, ar putea fi autorul unui document, codificarea caracterelor unui simplu document text, sau un control .

**Sisteme de gestiune** **de fisiere ierarhice**

Sistemul de gestiune de fisiere ierarhic a fost un interes de cercetare timpuriu al lui Dennis Ritchie ceea ce a dus la faima Unix-ului. Implementarile anterioare s-au limitat doar la cateva nivele, in special implementarea IBM, chiar si la primele lor baze de date cum ar fi IMS . Dupa succesul Unix-ului, Ritchie a extins conceptul de sistem de gestiune de fisiere pentru fiecare obiect in dezvoltarile viitoarelor lui sisteme de operare, cum ar fi Plan 9 si Inferno .

**Facilitati**

Sistemele de gestiune de fisiere traditionale ofera abilitatea de a crea, muta si sterge fisiere sau directoare. Le lipsesc facilitatile de a crea legaturi suplimentare la un director (*hard links* in Unix ), redenumirea linkurilor parinte (".." in sisteme gen Unix), si de a crea legaturi bidirectional la fisiere.

Sistemele de gestiune de fisiere traditionale, de asemenea, ofera abilitatea de a trunchia, adauga, crea, muta, sterge si modifica fisierele in loc. Ele nu ofera facilitati pentru adauga sau trunchia de la inceputul unui fisier, sa nu mai vorbim de inserare arbitrara sau stergerea dintr-un fisier. Operatiile prevazute sunt foarte asimetrice si lipsite de generalitate pentru a fi utile in contexte neasteptate. De exemplu, pipe-uri intre procese in Unix, trebuiesc implementate in afara sistemului de gestiune de fisiere, deoarece conceptul de *pipe* nu ofera trunchiere de la inceputul fisierelor.

**Acces securizat**

Accesul securizat la operatiile sistemului de gestiune de fisiere de baza se poate baza pe un sistem de liste de control de acces sau de capabilitati . Cercetarile au aratat ca listele de control de acces sunt dificil de securizat in mod corespunzator, motiv pentru care sistemele de operare de cercetare au tendinta de a utiliza capabilitati. Sistemele de gestiune de fisiere comerciale inca mai folosesc liste de control de acces.

**1.3.Tipuri de sisteme de gestiune** **de fisiere**

Pot fi clasificate in sisteme de gestiune de fisiere de disc, de retea sau cu un scop special.

**1.3.1. Sisteme de gestiune** **de fisiere de disc**

Un sistem de gestiune de fisiere de disc este un sistem de gestiune realizat pentru stocarea de fisiere pe un dispozitiv de stocare de date, cel mai frecvent o unitate de disc , care ar putea fi direct sau indirect conectat la computer. Exemple de sisteme de gestiune de fisiere pe disc includ FAT (FAT12, FAT16, FAT32, exFAT), NTFS , HFS si HFS + , HPFS , UFS , ext2 , ext3 , ext4 , btrfs , ISO 9660 , ODS-5 , Veritas File System , ZFS si UDF . Unele sisteme de gestiune de fisiere de disc sunt sisteme de jurnalizare sau de versiuni.

ISO 9660 si Universal Disc Format , sunt doua dintre cele mai comune formate care vizeaza Compact Disc-uri si DVD-uri . Mount Rainier este o extensie noua la UDF sustinuta de Linux 2.6 si Windows Vista care faciliteaza rescrierea DVD-urilor in acelasi mod ca la dischete.

**1.3.2. Sisteme de gestiune** **de fisiere flash** [1]

Un sistem de gestiune de fisiere flash este un sistem de gestiune proiectat pentru stocarea fisierelor pe dispozitive cu memorie flash. Acestea sunt ce in ce mai raspandite, deoarece numarul de dispozitive mobile este in crestere, in acelasi timp creste si capacitatea memoriilor flash.

In timp ce un sistem de gestiune de fisiere pe disc poate fi folosit pe un dispozitiv flash, aceasta este sub nivelul optim din mai multe motive:

* *stergerea blocurilor*: blocurile de memorie flash trebuie sa fie sterse in mod explicit, inainte ca acestea sa poate fi rescrise. Timpul necesar pentru a sterge blocuri poate fi semnificativ, deci este benefica stergea blocurilor neutilizate in timp ce aparatul este inactiv.
* *acces aleator*: sistemele de gestiune de fisiere de disc sunt optimizate pentru a evita seeking`uri de disc ori de cate ori este posibil, din cauza costului ridicat al seeking`ului. Dispozitive de memorie flash beneficiaza de lipsa latentei in acest caz.
* *wear leveling*: dispozitivele cu memorie flash au tendinta de a se uza atunci cand un singur bloc este suprascris in mod repetat; sistemele de gestiune de fisiere flash sunt proiectate sa disperseze uniform scrierea.

Sistemele de gestiune de fisiere cu “log” au multe din proprietatile bune ale sistemelor de gestiune de fisiere flash. Astfel de sisteme includ JFFS2 si YAFFS .

**1.3.3. Sisteme de gestiune** **de fisiere pe banda**

Un sistem de gestiune de fisiere pe banda este un sistem de gestiune si un format de banda proiectat pentru a stoca fisiere pe o banda intr-o forma simpla. Benzile magnetice sunt medii de stocare secventiale, care prezinta probleme la crearea si gestionarea eficienta a sistemului de gestiune de fisiere intr-un scop general. IBM a anuntat recent si a pus la dispozitie ca sursa deschisa un nou sistem de gestiune de fisiere pentru banda numit LTFS "Linear Tape File System" sau "Long Term File System". LTFS permite crearea de fisiere direct pe banda si utilizarea lor ca pe o unitate de disc obisnuita.

**1.3.4. Sisteme de gestiune** **de fisiere cu baze de date**

Un nou concept de gestionare a fisierelor este conceptul unui sistem de gestiune de fisiere bazat pe baze de date. In loc de, sau in plus fata de gestionarea structurata ierarhic, fisierele sunt identificate prin caracteristicile lor, ca tip de fisier, subiect, autor, sau metadate similare.

**1.3.5. Sisteme de gestiune** **de fisiere tranzactionale**

Fiecare operatiune pe disc, poate implica modificarea unui numar de diferite fisiere si structuri de disc. In multe cazuri, aceste modificari sunt legate intre ele, in sensul ca este important ca toate acestea sa fie executate in acelasi timp. De exemplu, in cazul unei banci care trimite bani catre o alta banca in format electronic, computerul bancii va "trimite" instructiuni de transfer la cealalta banca si va actualiza, de asemenea, propriile inregistrari pentru a indica faptul ca transferul a avut loc. Daca din vreun motiv oarecare calculatorul este avariat inainte de a fi avut sansa sa-si actualizeze propriile evidente, atunci la resetare, nu va fi nici o inregistrare de transfer, dar bancii ii vor lipsi niste bani.

Procesarea tranzactiilor introduce o garantie ca, in orice moment in care el functioneaza, o tranzactie poate fi finalizata complet sau recuperata complet (desi nu neaparat amandoua in acelasi moment). Aceasta inseamna ca, daca exista un accident sau pana de curent, dupa resetare, statutul inregistrat va fi consecvent. (fie banii vor fi transferati, fie nu vor fi transferati, dar nu vor fi niciodata pierduti "in tranzit").

Sistemele de gestiune de fisiere cu jurnalizare sunt o tehnica folosita pentru a introduce consecventa la nivelul tranzactional al structurilor sistemelor de gestiune de fisiere.

**1.3.6. Sisteme de gestiune** **de fisiere de retea** [2]

Un sistem de gestiune de fisiere de retea este un sistem de gestiune care actioneaza ca un client pentru un protocol de acces la distanta, oferind acces la fisierele de pe un server. Exemple de sisteme de gestiune de fisiere de retea includ clienti pentru protocoale NFS , AFS , SMB, si clienti ca sisteme de gestiune de fisiere pentru FTP si WebDAV .

**1.3.7. Sisteme de gestiune** **de fisiere comune de disc** [3]

Un sistem de gestiune de fisiere comune este unul in care un numar de masini (de obicei servere), toate au acces la acelasi subsistem de pe un disc extern (de obicei un SAN). Sistemul de gestiune de fisiere arbitreaza accesul la acest subsistem, previne coliziuni de scriere. Exemplele includ SFP de la Red Hat , GPFS de la IBM, si SFS de la DataPlow.

**1.3.8. Sisteme de gestiune** **de fisiere cu un scop special** [4]

Un sistem de gestiune de fisiere cu un scop special este, practic orice sistem de gestiune de fisiere care nu este un fisier de sistem de disc sau de retea. Aceasta include sisteme in care fisierele sunt organizate dinamic de software , destinate unor scopuri, cum ar fi comunicarea dintre procesele calculatorului sau spatiul fisierelor temporare.

Sisteme de gestiune de fisiere cu un scop special sunt cel mai des folosite de sistemele de operare de fisiere-centrice, cum ar fi Unix. Exemple includ sistemul procfs (/ proc) folosit de unele variante de Unix, care acorda acces la informatii despre procese sau alte caracteristici ale sistemului de operare.

Sateliti spatiali de explorare, cum ar fi Voyager I si II, foloseau sisteme de gestiune de fisiere speciale bazate pe benzi digitale. Acum, astfel de “nave” ca Cassini-Huygens utilizeaza un sistem de gestiune de fisiere ce lucreaza in timp real sau sistem de gestiune de fisiere cu influente RTOS. Roboteii trimisi pe Marte sunt un exemplu de astfel de sisteme de gestiune de fisiere RTOS, importante in acest caz deoarece acestea implementeaza memoria flash .

**Bibliografie:**

[1] http://en.wikipedia.org/wiki/Flash\_file\_system

[2] http://en.wikipedia.org/wiki/Network\_file\_system

[3] http://en.wikipedia.org/wiki/Shared\_disk\_file\_system

[4] http://en.wikipedia.org/wiki/Special\_file\_system

Referinte generale:

Jonathan de Boyne Pollard - "Disc and volume size limits"

IBM - "OS/2 corrective service fix JR09427"

Linux-NTFS Wiki (linux-ntfs.org) - NTFS Information, Linux-NTFS Project

Apple Computer Inc. - "Technical Note TN1150: HFS Plus Volume Format"

Brian Carrier, Addison Wesley – “File System Forensic Analysis”

http://ro.wikipedia.org/wiki/Sistem\_de\_fișiere

**2. GESTIUNEA SPATIULUI DE MEMORIE EXTERNA**

**ORGANIZAREA DATELOR IN MEMORIA EXTERNA**

(Banciu Alexandru)

**2.1. Evoluţia organizării datelor**

**2.2. Organizarea datelor în fişiere**

**2.3. Modele de structurare a datelor în baze de date**

**2.1. Evoluţia organizării datelor**

*Organizarea datelor* înseamnă definirea şi structurarea datelor în memoria internă sau în memoria externă (în colecţii) şi stabilirea legăturilor între date, conform unui model de date.

Bazele de date au evoluat din fişiere. Sintetizând, *evoluţia* organizării datelor în memoria externă a avut în vedere căteva *aspecte:*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Aspecte    Etape(1) | Mod de organizare  a datelor (2) | Structura de date  (3) | Mod de prelucrare (4) | Redundanţă  (5) | Software utilizat  (6) |
| 1. Înainte de 1965 | Fişiere secvenţiale | Logică coincide cu fizică | Pe loturi (batch) | Mare, necontrolat | Operaţii simple de I/E (limaje asamblare şi universal) |
| 2. Anii ‘60 | Fişiere secvenţiale, indexate, directe | Logică şi fizică | Loturi, on-line | Mare, necontrolat | Chei simple de acces (limbaje universale) |
| 3. Anii ‘70 | Baze de date arborescente, reţea | Logică, fizică, conceptuală | Loturi, conversaţi-onal | Scade, controlat | Chei multiple de acces, legături între date, protecţia (SGBD) |
| 4. Sfărşitul anilor ’70 până acum | Baze de date relaţionale | Logică, fizică, conceptuală | Conversaţi-onal, interactiv | Mică, controlat | Limbaje de regăsire, protecţie, concurenţă (SGBD) |
| 5. Sfârşitul anilor ’80 până acum | Baze de date orientate obiect | Logică, fizică, conceptuală | Interactiv | Minimă, controlat | Limbaje din programarea OO (SGBD) |

*Notă. Necesitatea* organizării datelor în memoria externă rezultă analizând câteva criterii de comparaţie dintre memoria internă şi cea externă:

|  |  |  |
| --- | --- | --- |
| CRITERIU | MEM. INTERNĂ | MEM. EXTERNĂ |
| Cost | Mare | Mic |
| Viteză | Mare | Mică |
| Capacitate(volum date) | Mică | Mare |
| Persistenţă | Nu | Da |
| Organizare date | Variabile, constante, masive, pointeri etc. | Fişiere, baze de date |

2.2. Organizarea datelor în fişiere

*Fisierul* este o colecţie organizată de date unite după criterii comune calitative, de prelucrare şi scop.

Toate limbajele de programare universale lucrează cu această noţiune, pentru organizarea datelor în memoria externă. Sistemele de bază de date lucrează cu această noţiune şi în plus o dezvoltă în noţiunea de bază de date.

Prezentăm, în continuare, câteva noţiuni fumdamentale utilizate în organizarea datelor în fişiere. De aceste noţiuni se ţine seama şi la baze de date.

1. Caracteristici ale unui fişier:

* *actualizarea* se referă la trei operaţii : adaugarea, modificarea, ştergerea de înregistrări;
* *natura* datelor din fişier trebuie să fie omogenă (să se refere la aceeaşi entitate din lumea reală);
* *prelucrarea* datelor din fişier se referă la tipul şi frecvenţa operaţiilor efectuate pe înregistrări;
* *volumul* de date din fişier (se măsoară în număr de octeţi).

1. Structura unui fişier:

* *partea de identificare* este dată de etichetele plasate la începutul şi sfârşitul fişierului;
* *partea de date* este colecţia omogenă de date ce aparţin aceleaşi entităţi din lumea reală structurată astfel:

fişier înregistrări câmpuri valori.

1. Modul de organizare reprezintă modul de dispunere a înregistrărilor pe suportul fizic şi presupune reguli de memorare a datelor.

*Categorii* de moduri de organizare a fişierelor:

a) Standard

* + Este cea mai veche şi există pe toate tipurile de calculatoare.
  + Înregistrarea este formată dintr-un şir de caractere dispus pe o linie acceptată de periferic.
  + Toate limbajele recunosc fişiere standard de intrare şi ieşire.

b) Clasică (elementară)

* + Organizarea se face pe medii magnetice sau optice.
  + *Tipuri:*
* SECVENŢIALĂ
  + Înregistrările sunt dispuse în fişier una după alta fără nici o ordine prestabilită.
  + Localizarea unei înregistrări se face prin parcurgerea tuturor înregistrărilor anterioare ei (secvenţial).
  + Toate sistemele de operare şi limbajele de programare acceptă organizarea secvenţială.
* RELATIVĂ
  + - * Înregistrările sunt dispuse în fişier una după alta şi numerotate (de către sistem) de la 0 sau 1 la câte sunt (număr de realizare)
      * Localizarea unei înregistrări se poate face secvenţial sau direct prin numărul de realizare.
* INDEXAT-SECVENŢIALĂ
  + - * Înregistrările sunt dispuse în fişier în ordine strict crescătoare după o cheie (face parte din înregistrare).
      * Cheia este unul sau mai multe câmpuri care identifică în mod unic o înregistrare.
      * Fişierului îi este ataşat o tabelă de indecşi care face legătura între valoarea cheii şi adresa fizică a înregistrării.
      * Localizarea unei înregistrări se poate face secvenţial dar şi direct prin cheie:
        + se compară cheia înregistrării căutate cu indecşii din tabela de index şi se localizează direct partea fizică a fişierului în care se află înregistrarea căutată;
        + în partea fizică localizată se face o căutare secvenţială a înregistrării dorite.

**c) Specială** (complexă)

* Se bazează pe modurile de organizare clasice.
* Sunt utilizate în baze de date şi în sisteme de fişiere.
* *Tipuri* (câteva):
* PARTIŢIONAREA
  + - Înregistrările din fişier sunt grupate în partiţii sub un nume.
    - În cadrul unei partiţii înregistrările sunt organizate secvenţial.
    - Se utilizează pentru bibliotecile de programe
* MULTIINDEXAREA
  + - Este o extindere a indexării prin utilizarea mai multor chei alese de programator.
    - Spaţiul ocupat este mai mare.
    - Se utilizează pentru fişiere care necesită regăsiri intense multicriteriale.
* INVERSĂ
  + - Presupune existenţa a două fişiere: de bază şi invers.
    - Fişierul de bază conţine datele propriu-zise şi are organizare secvenţială. El este fişierul în care se caută.
    - Fişierul invers este construit din cel de bază (printr-o tehnică de inversare) şi are organizare relativă. El este fişierul prin intermediul căruia se caută.
    - Spaţiul ocupat necesar este cam de 3,5 ori mai mare faţă de cât ocupă fişierul de bază.

4) Modul de acces reprezintă modul în care se determină locul ocupat de o înregistrare într-un fişier şi depinde de modul de organizare.

*Tipuri* de moduri de acces pentru fişiere:

* SECVENŢIAL presupune că localizarea unei înregistrări se face prin parcurgerea tuturor înregistrărilor care o preced.
  + - Este permis accesul secvenţial pentru toate tipurile de fişiere.
    - Se recomandă pentru fişierele din care sunt necesare, la o prelucrare, peste 50% din numărul total de înregistrări.
    - Pentru optimizare se recomandă ordonarea fişierului.
* DIRECT presupune că localizarea unei înregistrări se face cu ajutorul unei chei definite de programator.

*Notă.* Accesul direct se poate face după numărul de realizare sau după o valoare a cheii.

* DINAMIC presupune că la o singură deschidere de fişier se pot localiza, alternativ şi repetat, înregistrări în acces secvenţial şi direct.

*Notă.* Criterii de comparaţie pentru organizarea datelor în fişiere şi baze de date sunt prezentate în Anexa 2.

**2.3. Modele de structurare a datelor în BD**

**Informaţia***,* care se reprezintă în calculator în memoria internă sau externă, se poate defini structural după schema:

are ca are ca are ca

informaţia entitatea atributul valoarea

obiect proprietate măsură

**Modelul de structură** se referă la descrierea tuturor atributelor unei entităţi în interdependenţă.

Valorile atributelor se materializează prin *date*, care dau o reprezentare simbolică a informaţiilor.

**Modelul de date** este ansamblul de concepte şi instrumente necesar pentru a realiza structura colecţiilor de date (schema) (fig. 2.1.).

Modelul de date este *compus* din:

* concepte;
* un formalism pentru a descrie datele (structura de date);
* un ansamblu de date pentru a le manipula (datele).

CONCEPTE STRUCTURA DE DATE

(formalism de descriere)

DATELE

(ansamblul de manipulat)

Fig 2.1. Modelul de date

În literatura de specialitate [RICC01], [RAGE00] sunt prezentate trei tipuri de modele de date pentru baze de date. Prezentăm, pe scurt, aceste tipuri, împreună cu câteva caracteristici pentru fiecare:

* *modelul conceptual* (la analiză)
  + descrie sistemul în termeni pe înţelesul utilizatorului;
  + se foloseşte pentru specificaţia iniţială a datelor şi pentru comunicarea cu utilizatorii;
  + este un mod de stabilire a unei legături între dezvoltatori şi utilizatori.
* *modelul logic* (la proiectare)
  + specifică structura bazei de date (colecţiile de date şi legăturile dintre ele);
  + se scrie într-o formă care să poată fi folosită pentru crearea bazei de date;
  + se foloseşte pentru definirea şi întreţinerea bazei de date de către SGBD şi pentru formularea cerinţelor de regăsire de către utilizatori.
* *modelul fizic* (sarcina SGBD)
  + descrie modul în care modelul logic al datelor va fi reprezentat la stocare, în memoria externă (fişiere, indecşi, discuri etc.);
  + de obicei, este generat automat, de către SGBD, pornind de la modelul logic.

**Schema** *[conceptuală]* reprezintă descrierea fenomenelor din realitatea înconjurătoare prin entităţi şi atribute (tipurile de date), împreună cu toate corelaţiile (legăturile) dintre ele (constrângerile).

Definirea schemei este o activitate de *modelare* pentru că traduce în termeni abstracţi entităţile lumii reale.

Schema BD *se reprezintă* cu ajutorul unui model de date implementat prin intermediul unui SGBD adecvat.

Trebuie să se facă *distincţie clară* între structura bazei de date (schema, de exemplu: *stud(cod:N,4; nume:C,15)* ) şi conţinutul ei (instanţa, de exemplu: *22 Nedelcu Anda*). Schema şi instanţele sunt stocate în baza de date, şi anume prima în docţionar, iar cea de-a doua în datele propriu-zise.

În cele ce urmează ne vom ocupa, în cea mai mare parte, de modelul logic de date pentru baze de date.

**Elementele** (componentele) oricărui model de date pentru baze de date sunt:

1. Definirea structurii modelului (partea structurală):
   * + definirea entităţilor şi a atributelor asociate;
     + definirea legăturilor (asociarea) dintre entităţi.

Asocierea poate fi de *tipul* :

* + - unu la unu (1:1)
    - unu la mulţi (1:M)
    - mulţi la mulţi (M:N)
    - neprecizat explicit

Definirea structurii de date se face cu un LDD (Limbaj de Descriere a Datelor) dintr-un SGBD.

1. Operatorii modelului (partea de manipulare) care acţionează asupra structurilor de date, dar şi asupra datelor, pentru operaţii de prelucrare (compunere, actualizare etc.).

Operatorii se implementează cu ajutorul unui LMD (Limbaj de Manipulare a Datelor) dintr-un SGBD.

1. Regulile de integritate (partea de coerenţă) sunt restricţii stabilite la descrierea datelor, care le asigură acestora menţinerea corectitudinii şi dau logica modelului.

Restricţiile se implementează cu un LDD din SGBD.

**Tipuri** de modele logice de date pentru BD:

* *fundamentale*: ierarhice, reţea, relaţionale, orientate obiect;
* d*erivate* (extinse din cele fundamentale): distribuite, multimedia etc.

Prezentăm, în continuare, principalele caracteristici pentru modelele de date fundamentare (de bază) pentru baze de date.

**Modelul ierarhic**

1. Definirea structurii modelului ierarhic.
2. **Definirea entităţilor** se face prin noţiune de *tip de înregistrare* (clasă de entităţi), care este formată din *caracteristici* (câmpuri).

*Realizarea* (instanţa) unui tip de înregistrare este dată de ansamblul valorilor pentru câmpurile acesteia (înregistrarea)..

1. **Definirea legăturilor** dintre entităţi se face *fizic* şi conduce la structura de *tip ierarhic* (arborescent) reprezenată sub forma unei *diagrame* (fig. 2.2.).

nivel 0 TIP\_ÎNREG 0

nivel 1 TIP\_ÎNREG 11 TIP\_ÎNREG 12 TIP\_ÎNREG 13

nivel 2 TIP\_ÎNREG 21 TIP\_ÎNREG 22 TIP\_ÎNREG 23

. . .

Fig. 2.2. Diagrama de structură ierarhică

**Caracteristici** ale structurii ierarhice (arborescente):

* Fiecare nod corespunde unui tip de înregistrare şi fiecare drum corespunde unei legături (asocieri).
* Orice acces la un nod se face prin vârful ierarhiei, numit *rădăcină*, pe o singură cale.
* Un nod subordonat (copil) nu poate avea decât un singur superior (părinte).
* Un superior poate avea unul sau mai mulţi subordonaţi.
* Legătura copil-părinte este doar de tip 1:1 (la o realizare copil corespunde o singură realizare părinte).
* Legătura părinte-copil poate fi de tip 1:1 sau 1:M.
* În structură există un singur nod rădăcină şi unul sau mai multe noduri dependente situate pe unul sau mai multe niveluri.
* ierarhie de tipuri de înregistrări se numeşte *tip arbore*.
* *realizare* a unui tip arbore este formată dintr-o singură realizare a tipului de înregistrare rădăcină împreună cu o mulţime ordonată formată din una sau mai multe realizări ale fiecărui tip de înregistrare de pe nivelurile inferioare.
* Ordonarea realizărilor dintr-un arbore conduce la o *secvenţă ierarhică*.

La un acces, întreaga secvenţă ierarhică este disponibilă automat.

1. Operatorii modelului ierarhic

* *Localizarea unui arbore* în BD: se localizează o realizare a tipului de înregistrare rădăcină.
* *Trecerea de la un arbore la altul* în BD: se trece de la o realizare a tipului înregistrare rădăcină (secvenţă ierarhică) la o altă realizare a aceluiaşi tip înregistrare rădăcină.
* *Trecerea de la o realizare* (înregistrare) *la alta* într-un arbore (secvenţă ierarhică) : se poate face trecerea pe acelaşi nivel sau pe niveluri diferite între tipuri de înregistrări legate între ele.
* *Actualizarea într-un arbore* : adăugarea, modificarea sau ştergerea unei înregistrări. Operaţia este greoaie şi consumatoare de resurse calculator (spaţiu şi timp) pentru că antrenează automat toate înregistrările din arbore care se înlănţuie cu înregistrarea actualizată.

*Notă*. Operatorii de mai sus sunt la nivel de înregistrare; acţionează pe o înregistrare şi produc tot o înregistrare.

Există şi operatori la nivel de mulţime de înregistrări care se implementează în LMD mult mai greu.

1. Restricţiile de integritate ale modelului ierarhic.

* realizare subordonat este totdeauna asociată unei singure realizări superior.
* Dacă un tip de înregistrare nu are realizări atunci nici tipurile înregistrări descendente nu au realizări.

*Notă*. Efectele restricţiilor de integritate sunt:

* complicarea operaţiei de actualizare;
* dacă o realizare a unui nod subordonat trebuie să fie asociată cu mai multe realizări ale nodului părinte atunci ea trebuie multiplicată (creşte redundanţa);
* dau logica şi corenţa modelului arborescent.

**Caracterizarea** generală a modelului ierarhic.

1. Modelul a fost *propus* de către IBM şi a fost primul utilizat pentru BD.
2. *Structura* modelului este simplă (graf orientat) familiară specialiştilor în informatică.
3. *Implementarea* modelului se face fizic şi condiţionează performanţele BD:

* prin pointeri, utilizând diferite metode (liste simplu înlănţuite, vectori de pointeri, chei primare etc.);
* secvenţial, fiecare realizare a unui arbore corespunde unei înregistrări (logice) dintr-un fişier secvenţial.

1. *Aplicabilitatea* modelului se regăseşte cu succes în tehnologia construcţiilor de maşini, dar şi în alte domenii.
2. *Limitele* modelului ierarhic:

* numărul de ierarhii posibile creşte combinatoric cu numărul înregistrărilor, în legătura părinte-copil;
* actualizarea datelor este groaie şi consumatoare de resurse calculator;
* nivelul logic nu este separat clar de cel fizic (exemplu: indecşii pot fi structuri logice dar şi fizice);
* nu se poate realiza legătura de tip M:N

1. *Exemplu* de SGBD ierarhic este IMS (Information Management System) realizat de către IBM.
2. *Baza de date ierarhică* este o mulţime ordonată de realizări ale unui tip arbore.

**Modelul reţea**

1. Definirea structurii modelului reţea.

Definirea entităţilor se face prin noţiunea de *tip de înregistrare*, care este formată din *caracteristici* (câmpuri).

*Realizarea* (instanţa) unui tip de înregistrare este dată de ansamblul valorilor pentru câmpurile acestuia (înregistrarea).

Definirea legăturilor dintre entităţi se face *fizic* şi conduce la o structură de tip *reţea*, reprezentată sub forma unei *diagrame* (fig. 2.3.) numită şi schema (conceptuală).

TIP\_ÎNREG 1 TIP\_ÎNREG 2

TIP\_ÎNREG 3 TIP\_ÎNREG 4

TIP\_ÎNREG 5 TIP\_ÎNREG 6

. . .

Fig. 2.3. Diagrama de structură reţea

**Caracteristici** ale structurii reţea :

* Un nod poate avea oricâţi superiori şi oricâţi subordonaţi.
* Pot exista mai multe tip de înregistrare rădăcină.
* La un subordonat se poate ajunge pe mai multe căi.
* Este o extensie a structurii arborescente în care graful nu mai este limitat.
* Legăturile în model pot fi 1:1, 1:M, M:N, ciclice.
* Pentru exprimarea legăturilor se foloseşte conceptul de *tip set*. Un tip set este format dintr-un singur tip de *nod proprietar* şi unul sau mai multe tipuri de *noduri membre* dependente de cel proprietar.
* *realizare* a tipului set este o colecţie de înregistrări care are o realizare proprietar şi un număr de realizări membre asociate.

1. Operatorii modelului reţea.

* *Localizarea* *unui tip set*: se identifică o realizare a unui nod proprietar.
* *Trecerea* de la o înregistrare la alta în cadrul unui set: proprietar-membru, membru- proprietar, membru-membru.
* *Actualizarea într-un set*: adăugarea, modificarea, ştergerea unor membri.
* *Actualizarea într-o reţea*: adăugarea, modificarea, ştergerea unor seturi.

*Notă*. Operaţiile de actualizare sunt greoaie şi mari consumatoare de resurse calculator. Aceasta, deoarece sunt antrenate întotdeauna înregistrările dintr-unul sau mai multe seturi.

* *Localizarea unei înregistrări* pe baza valorii unui câmp (cheie).

1. Restricţiile de integritate ale modelului reţea.

* înregistrare nu poate fi membră a două realizări ale aceluiaşi tip set. Înregistrarea se va multiplica.
* înregistrare poate să aparţină mai multor tipuri set prin multiplicare.
* Un tip de înregistrare poate fi nod proprietar într-un set şi nod membru în alt set.
* Un set poate avea un singur nod proprietar.

*Notă.* Efectele restricţiilor de integritate sunt:

* dau logica şi coerenţa modelului reţea;
* complică operaţia de actualizare;
* introduce o redundanţă controlată.

**Caracterizarea** generală a modelului reţea.

1. A fost *propus* de CODASYL ca soluţie pentru a se elimina limitele modelului ierarhic.
2. Se poate utiliza pentru domenii variate din lumea reală, deoarece permite reprezentarea unor *structuri* *complexe*.
3. *Implementarea* modelului se face fizic şi condiţionează performanţele BD:

* prin pointeri structuraţi în liste înlănţuite, de diferite tipuri (owner, prior, next);
* prin hartă de biţi (matrice cu înregistrări, iar la intersecţie sunt legăturile: 1-da, 0-nu).

1. *Limitele* modelului reţea:
   * complexitatea modelului îl face dificil de implementat;
   * legăturile din model nu sunt întotdeauna foarte clare;
   * prelucrarea înregistrărilor se face secvenţial;
   * actualizarea este greoaie şi consumatoare de resurse calculator;
   * reproiectarea este dificilă datorită complexităţii ridicate.
2. *Exemple* de SGBD reţea : IDMS (Integrated Database Management System), Socrate.
3. *Baza de date reţea* este o mulţime oarecare de tipuri de înregistrări structurate pe tipuri set.

**Modelul relaţional**

1. Definirea structurii modelului relaţional.

a) **Definirea entităţilor** se face sub forma unor tablouri bidimensionale numite *tabele* sau *relaţii* de date.

*Conceptele* utilizate sunt:

DOMENIU este un ansamblu de valori caracterizat printr-un nume.El poate fi explicit (se enumeră valorile posibile, de exemplu D1:{M, F}) sau implicit (se precizează proprietăţile valorilor, de exemplu D1:{a/a∈N}).

TABELA/RELAŢIA este un subansamblu al produsului cartezian al mai multor domenii, caracterizat printr-un nume.

ATRIBUTUL este coloana unei tabele, caracterizată printr-un nume

TUPLUL este linia dintr-o tabelă şi nu are nume.

*Notă*. Ordinea liniilor (tupluri) şi coloanelor (atribute) dintr-o tabelă nu trebuie să prezinte nici-o importanţă.

SCHEMA TABELEI este numele tabelei, urmat între paranteze rotunde de lista atributelor, iar pentru fiecare atribut se precizează domeniul asociat.

CHEIA este un atribut sau un ansamblu de atribute care au rolul de a identifica un tuplu dintr-o tabelă.

*Tipuri* de chei: primare/alternate, simple/comune, externe.

SCHEMA RELAŢIONALĂ este schema tabelei + cheile + restricţiile de integritate.

b) **Definirea legăturilor** dintre entităţi se face *logic* construind asocieri între tabele cu ajutorul unor atribute de legătură.

Legăturile se pot reprezenta sub forma unei *diagrame de structură* (fig. 2.4.) numită şi *schema* BD.

**Caracteristici** ale structurii relaţionale:

* + Atributele implicate în realizarea legăturilor se găsesc fie în tabelele asociate, fie în tabele distincte construite special pentru legături.
  + Atributul din tabela iniţială se numeşte *cheie externă* iar cel din tabela finală este *cheie primară*.
  + Legăturile posibile sunt 1:1, 1:M, M:N.
  + Potenţial, orice tabelă se poate lega cu orice tabelă, după orice atribute.
  + Legăturile se stabilesc la momentul descrierii datelor (prin LDD) cu ajutorul restricţiilor de integritate.

Practic, se stabilesc şi legături dinamice la momentul execuţiei.

1. Operatorii modelului relaţional.

a) Operatori din **algebra relaţională**:

* standard: selecţia proiecţia, joncţiunea, reuniunea, diferenţa, produsul cartezian, intersecţia, diviziunea;
* extensii : complementarea, descompunerea (splitarea) etc.

*Notă.* Operatorii algebrei relaţionale acţionează la nivel de tabelă în expresii relaţionale. Operanzii sunt tabele iar rezultatul este întotdeauna o tabelă.

b) Operatorii din **calculul relaţional**:

* orientaţi pe tuplu: conectivele (conjuncţia ∧, disjuncţia ∨, negaţia ⎤ ), cuantificatorii (existenţial ∃, universal ∀);
* orientaţi pe domeniu: idem ca mai sus.

*Notă*. Pentru operatorii calcului relaţional operandul poate fi tuplu sau domeniu.

1. Restricţiile de integritate ale modelului relaţional.

a) **Structurale** sunt cele care se definesc prin compararea unor valori din tabele:

* *cheie unică* : într-o tabelă nu trebuie să existe mai multe tupluri cu aceeaşi valoare pentru ansamblul cheie;
* *referenţială*: într-o tabelă T1 care referă o tabelă T2, valorile cheii externe trebuie să figureze printre valorile cheii primare din T2 sau să ia valoarea NULL (neprecizat);
* *entităţii* : într-o tabelă, atributele din cheia primară nu trebuie să ia valoarea NULL.

*Notă.* Cele trei restricţii de mai sus sunt *minimale*.

Pe lângă acestea, există o serie de alte restricţii structurale care se referă la dependenţele dintre date: funcţionale, multivaloare, joncţiune etc. (sunt luate în considerare la tehnicile de proiectare BD relaţionale).

b) **Semantice** sunt cele care se definesc prin comportamentul datelor şi ţin cont de valorile din BD:

* restricţia *de domeniu* : domeniul corespunzător unui atribut dintr-o tabelă trebuie să se încadreze între anumite valori;
* restricţii *temporare* : valorile anumitor atribute se compară cu nişte valori temporare (rezultate din calcule etc.).

*Notă*. Restricţiile semantice fiind foarte generale se gestionează fie la momentul descrierii datelor (de exemplu prin clauza CHECK), fie în afara modelului la momentul execuţiei (de exemplu prin instrucţiunea IF) .

**Caracterizarea** generală a modelului relaţional.

1. A fost *propus* de către IBM şi a revoluţionat reprezentarea datelor în BD făcând trecerea la o nouă generaţie (a doua).
2. Modelul este *simplu*, are o solidă *fundamentare teoretică* fiind bazat pe teoria seturilor (ansamblurilor) şi pe logica matematică.
3. Pot fi reprezentate toate tipurile de structuri de date de mare complexitate, din *diferite domenii* de activitate.
4. *Implementarea* modelului se face logic prin atribute având rol de chei.
5. *Limitele* modelului relaţional:
   * prea marea simplitate a modelului îl face dificil de aplicat pentru noile tipuri de aplicaţii (multimedia, internet etc.);
   * nu asigură o independenţă logică totală a datelor de aplicaţie;
   * pentru aplicaţii de volum şi complexitate foarte mari nu mai face faţă;
   * poate introduce o redundanţă prea mare (la proiectare prin tehnica de normalizare).
6. *Baza de date relaţională* este un ansamblu de tabele prin care se reprezintă atât datele cât şi legăturile dintre ele.

**Bibliografie: Cursuri Baze de date**

**3. Sisteme de gestiune de fisiere Windows (FAT, NTFS, etc.)**

(Pavaloaia Oriana)

**3.1. Concepte fundamentale : exemple si comparatii Linux si Windows**

**3.1.1. Definitii.Concepte**

Structura de date ce include un volum mare de informatii procesate omogen se numeste fisier.Una din abstractiile fundamentale in lumea sistemelor de operare , are o forma standardizata si este folosit pentru a stoca informatiile necesare functionarii sistemului de operare si interactiunii cu utilizatorul.

Modul de organizare a fisierelor si de prezentare a acestora utilizatorului se realizeaza cu ajutorul unui sistem de fisiere. Sistemul de fisiere are o structura ierarhica de directoare si fisiere , incepand cu un director radacina. Cu ajutorul unei cai in care sunt prezentate toate intrarile de pana atunci , se realizeaza localizarea unui fisier in sistemul de fisiere.

**3.1.2. Reprezentarea partitiilor**

Windows are un sistem diferit fata de sistemul de fisiere adoptate de SO Linux.Exista o singura structura directoare in Linux.Incepe cu directorul (root) reprezentate prin simbolul “/” , dupa care se extinde in sub-directoare.In Windows acest lucru este tratat diferit: utilizatorul are mai multe resurse de tip disc, notate A:-Z(“driver letter”) pentru FD fiind folosit A,B:, fiecare fiind radacina unui arbore separat. Toate partitiile din Linux sunt plasate sub directorul radacina “root” in care se “mounteaza” sub denumirea unor directoare. In Windows cel mai aproape de “radacina” este partitia “C:”. O fractiune a HDD-ului unui calculator a carei marime o selecteaza utilizatorul , aceasta fiind cuprinsa intre 1-100% din marimea hard-diskului se numeste partitie.In functie de necesitatile utilizatorului , un calculator poate avea una sau mai multe partitii.

Diversele partitii existente in Windows , sunt detectate la bootare si li se ofera o litera din alphabet (C-Z). In Linux , sistemul nu stie de existent acestora daca nu se mounteaza o partitie sau un dispozitiv.Ofera o mare flexibilitate in sensul ca se pot ascunde anumite portiuni din HDD sau in functie de necesitatea utilizatorului diverse componente, e un plus pentru securitate si ergonomie dar nu este o metoda usoara de acces pentru incepatori.Modul de abordare al sistemului de fisiere unificat din Linux are si alte avantaje.De exemplu majoritate fisierelor executabile le contine directorul “/usr”. Linux ne permite sa mountam acest director pe alta partitie sau chiar pe alt calculator din retea.Deoarece directorul “/usr” ca aparea ca un director local , sistemul nu va detecta nici o incompatibilitate.Acest lucru in Windows nu este posibil, de exemplu mutarea directorului “Program Files” pe alta partitie nu pastreaza o functionalitate totala.

La prima vedere este un lucru minor , dar piesa importanta in functionabilitate este ca Linux foloseste slash “/” pentru despartirea cailor catre fisiere , spre deosebire de back-slashul “\” folosit de Windows.Pentru un obisnuit cu Windows aceasta reacomodare poate fii destul de dificila , Durand ceva vreme pana la acomodare.Mai mult , spre deosebire de Windows unde nu conteaza daca denumirea fisierului este introdusa cu litere mari sau litere mici , Linux este un sistem “case sensitive” adica face diferenta intre dimensiunile literelor folosite.

Fisierele in Linux sunt identificate prin i-number(index dintr-un sir de i-noduri).Fiecare fisier are un singur i-node care contine.

1.identificatorul utilizatorului ce este proprietarul fisierului

2.tipul fisierului (pipe sau special , director , obisnuit)

3.drepturile de acces

4.data si ora ultimei modificari efectuate asupra i-node-ului precum si timpul ultimului acces si al ultimei modificari

5.numarul de legaturi(a se vedea comanda unlink)

6.adresele sectoarelor de pe HDD ce contin datele fisierului

7.lungimea fisierului in octeti

**3.1.3. Sisteme de fisiere Windows (FAT&NTFS)**

Datorita implementarii in sistemul de operare Microsoft , sistemul de fisiere FAT ( File Allocation Table) a devenit faimos.Sistemul de fisiere avea si are mai multe defecte.Sistemele (FAT12 si FAT16) sunt versiunile initiale , utilizau doar 11 caractere pentru numele fisierelor (opt pentru nume si 3 pentru extensie) si suportau o dimensiune limitata pentru HDD ( 250 Mb pentru FAT12 si 4Gb pentru FAT16).Versiunea FAT32 a rezolvat problema cu HDD marind limita la 2TB, iar extinderea limitei pentru numele fisierelor la 255 de caractere sa realizat cu ajutorul lui vFAT.O partitie FAT are nevoie din cand in cand de o defragmentare datorita naturii sale , deoarece blocurile ce compun un fisier pot fii imprastiate incetinind semnificativ operatiunile de citire. Avantajul sistemului de fisiere FAT este ca practic aproape toate sistemele de operare il suporta.Poate fii luata in considerare si utilizarea Linuxu-lui in combinative cu o partitie FAT deoarece poate ajuta la schimbul de date intre Linux si Windows sau poate mentine o versiune mai veche de Windows.O varianta de FAT este sistemul de fisiere UMSDOS, care permite instalarea (pe acceasi partitie si fara drivere suplimentare ) a unui sistem Linux (chiar daca acesta va avea prestatii ceva mai modeste fata de un sistem de fisiere native) cat si a unui sistem Windows.

Windows NT foloseste sistemul de fisiere NTFS (NT file system) care a fost apoi imbunatatit pentru Windows 2000,XP si Vista.Windows NT foloseste NTFS4 , in timp ce versiunea putin imbunatatita NTFS5 este folosit pentru Windows 2000 si XP.Diferentele dintre NTFS5 si NTFS4 sunt:

1.Encriptarea directoarelor si a fisierelor

2.Posibilitatea limitarii accesului diversilor useri la partitii impunandu-se o anumita marime a spatiului pe disk disponibil.

3.Recovery Console

4.Dynamic Volume Management

**3.1.4. Caracteristici NTFS**

Facilitati ale acestui sistem de fisiere:

1.Poate suporta partitii de pana la 2(puterea 64) bytes si foloseste adrese de disc de 64 de biti;

2.In numele fisierelor exista posibilitatea folosirii de caractere Unicode.Standardul de codificare si interpretare a datelor binare in format text se numeste Unicode , incluzand in varianta finala toate caracterele folosite in orice limba.Este proiectat pentru ca oricarui caracter din orice limba , de pe orice platform , sau program , sa ii corespunda un singur numar;

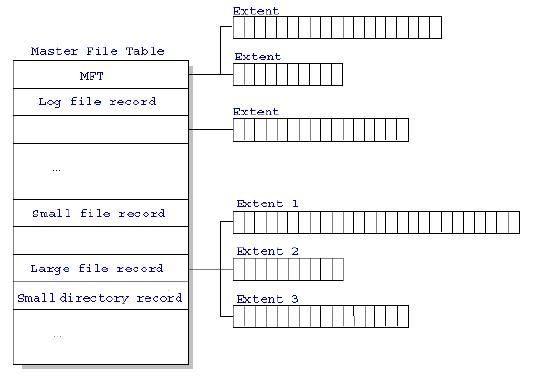
3.Folosirea numelor de fisiere de pana la 255 de caractere (inclusiv spatii si puncte);

4.Indexarea generala a fisierelor este permisa;

5.Ofera posibilitatea managementului dinamic al sectoarelor ;

6.Face distinctive intre litere mari si mici in cadrul numelor de fisiere ,folosindu-se de POSIX (Portable Operating System Interface ) permite crearea de “hard-link-uri” si pastreaza informatii de timp referitoare la fisier.Destinat managementului API (Application Programming Interface) este POSIX;

7. Utilizarea fisierelor cu seturi multiple de date este permisa.

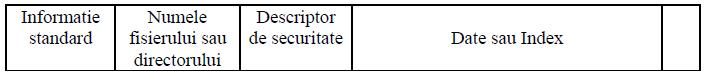


**Fig 1.1 Structura MFT**

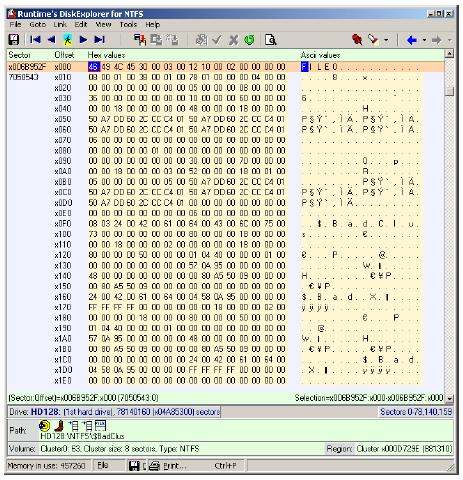
**3.1.5. MBR si MFT**

Odata cu bootarea sistemului se executa un program mic ce se numeste MBR ( “Master Boot Record”).De obicei el se afla pe primul sector al hard-diskului.Uitandu-se in tabelul de partitii , programul demareaza procesul de bootare si decide ce partitie este folosita pentru bootare.Apoi transfera controlul programului in sectorul de boot al partitiei , care va continua procesul de boot.MBR-ul , in DOS si in Windows , se creaza folosind comanda FDISK/MBR.El ocupa 512 byte.

O serie de fisiere sistem printre care si Master File Table (MFT) sunt create atunci cand in Windows este formatata o partitie pentru prima oara in sistem NTFS , si contine informatii despre toate fisierele si directoarele de pe volumul in cauza.Sectorul de boot este prima informatie pe o partitie NTFS si reprezinta sectorul 0 al partitie ce contine un cod de pornire al sistemului.In sectoarele 1-16 se regasesc alte informatii necesare programului de boot-are.MFT este primul fisier pe un volum NTFS.Exista cel putin o intrare in MFT pentru fiecare fisier de pe un volum NTFS. In MFT sunt pastrate toate informatiile despre un fisier ( nume , dimensiune , data creare, etc.).



**Fig 1.2 MFT record pentru un fisier sau director mic**



**Fig** **1.3 Structura grafica MFT**

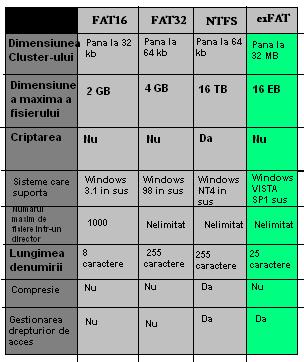
**3.1.9. Sistemul de fisiere exFAT**

exFAT (Extended FAT File System) dupa cum bine ii spune si numele este varianta imbunatatita a formatului FAT (FAT32). Acest sistem de administrarea a fisierelor se recomanda si se utilizeaza in special pe dispozitivele portabile cu o capacitate de stocare mare.

Cel mai mare plus pentru exFAT consta in faptul ca fisierele mai mari de 4GB (fisierele media de obicei) pot fi copiate fara a fi nevoie o impartire a lor (split). Daca nu stiati in formatul FAT32 un fisier nu poate avea mai mult de 4GB si o partitie pe hard disk putea avea maxim 32GB. Cu exFAT partitia unui hard disk poate avea pana la maximum 512TB.

In momentul de fata cel mai utilizat sistem de fisiere este NTFS, chiar si pe memoriile portabile, dar exista un inconveninet al acestui format si anume ca implementarea lui necesita o anumita parte de stocare din memorie.

Spre exemplu daca se utilizeaza exFAT pe un stick de 4GB este nevoie de doar 96 Kilobytes, iar pentru NTFS este nevoie de 47 Megabytes.



**Fig 1.4 Tabel comparativ intre sistemele de fisiere FAT**

**3.1.10. Sistemul de fisiere ZFS**

Creat in 2005 de catre Sun, ZFS (Zettabyte File System) este primul sistem de fisiere din lume pe 128 biti, permitand o capacitate de stocare de 18 miliarde de ori mai mare decat cea a sistemelor actuale pe 64 biti.

ZFS este dezvoltat de catre Sun si este publicat sub licenta open-source, fiind folosit in sistemul sau de operare Solaris si in proiectul OpenSolaris.Sistemul de fisiere ZFS are o serie de avantaje fata de sistemele traditionale, printre care integrarea de mecanisme care asigura integritatea datelor, suport pentru “storage pools”, permitand adaugarea de capacitate fara manipulare de volume si partitii, si crearea de snapshot-uri, care marcheaza modificarile prin care trece un fisier, permitand restaurarea unei stari anterioare, functionalitate anuntata deja in Leopard sub numele de Time Mschine.

Leopard va fi lansat oficial in octombrie anul acesta, dupa ce a fost amanat pentru a permite lansarea la timp a iPhone-ului. Dar detalii despre functionalitatea acestuia sunt asteptate la conferinta Apple WWDC (World Wide Developer Conference) de saptamana viitoare din San Francisco.

# 3.1.11. EFS (Encrypting File System)

# Sistemul de fisiere cu criptare (EFS) este o caracteristică Windows care se poate utiliza pentru a depozita informatiile de pe hard disk într-un format criptat. Criptarea este cea mai puternică protectie pe care Windows o furnizează pentru a ajuta la păstrarea în sigurantă a informatiilor.

# Unele caracteristici cheie ale EFS:

# 1. Criptarea este simplă; doar bifati o casetă de selectare în proprietătile fisierului sau folderului pentru a o activa.

# 2. Detineti controlul asupra persoanelor care pot citi aceste fisiere.

# 3. Fisierele sunt criptate când le închide si, dar sunt automat gata de utilizare atunci când le deschide si.

# 4. Dacă vă răzgândi si în ceea ce priveste criptarea unui fisier, debifati caseta de selectare în proprietătile fisierului.

**Bibliografie:**

http://en.wikipedia.org/wiki/NTFS

www.stst.elia.pub.ro

http://en.wikipedia.org/wiki/Fat

http://en.wikipedia.org/wiki/ZFTPServer\_Suite

http://en.wikipedia.org/wiki/Master\_boot\_record

4. Sisteme de gestiune de fisiere UNIX

(Sasu Alexandru)

**4.1. Generalitati. Deosebirea fata de Windows.**

Sistemul de fisiere din Linux este foarte diferit fata de cel din Windows astfel incat poate fi un mare impediment pentru utilizatorii care trec de la sistemele de operare Microsoft la un sistem Unix-based.

Spre deosebire de Windows, care structureaza discul in **partitii** care cuprind foldere(directoare), sub-foldere, fisiere, in Linux, totul porneste de la un director "**radacina"**, reprezentat prin "/" si se extinde prin sub-directoare. Astfel, Linux plaseaza toate partitiile sub directorul radacina in cadrul acestei ierarhii, prin "montarea" lor in cadrul unor anumite directoare. In Windows, discul este format din una sau mai multe partitii, notate de catre sistem prin litere, de la A la Z, unele din acestea fiind rezervate de Windows pentru floppy, pentru unitatile optice, etc. Cel mai aproape de directorul radacina din Linux se poate aprecia ca fiind partitia c: ( in cazul in care aici s-a instalat sistemul de operare).

Sub Windows, diversele partitii sunt detectate la bootare si li se atribuie o litera (drive letter). Sub Linux, daca nu "montezi" o partitie sau un dispozitiv, sistemul nu va sti despre existenta acesteia/acestuia. Poate nu pare cea mai simpla metoda de a oferi access la partitiile sistemului sau la dispozitive dar ofera foarte multa flexibilitate.

Acest tip de schema, de plan, cunoscut sub numele de sistem de fişiere unificat, ofera mai multe  
avantaje faţă de abordarea pe care Windows o utilizează.

**4.2. Crearea unui sistem de fisiere sub Linux**

Pentru a putea fi utilizat, un sistem de gestiune a fisierelor trebuie creat intai. Aceasta operatie se numeste formatare pe sistemele Windows sau Ms-DOS. Un proces similar are loc si pe sistemele Unix-based. Daca in Windows, partitiile au nume date prin litere, de la A la Z, pe Linux, partitiile au nume in functie de tipul hard-diskului. Astfel, exista doua tipuri de hard-diskuri:

* discuri **(E)IDE**  sunt denumite **had, hdb, hdc, hdd**;
* discuri **SCSI,** denumite **sda, sdb, sdc**, etc.

In functie de tipul discului, acesta poate fi partitionat in mai multe partitii, numerotate de catre sistem. Spre exemplu, un disc sda va fi partitonat in sda1, sda2, etc.

Fiecare disc poate fi *partitionat* in partitii care vor fi numerotate, de exemplu pentru discul **sda**, cu **sda1, sda2, sda3**, etc.

  Crearea unui sistem de fisiere se realizeaza intr-una din partitiile dorite. De exemplu, daca dorim sa cream un sistem de fisiere de tip **ex2fs** pentru *sistemul* **Linu**x, se utilizeaza *comanda*:

**mkfs  [parametri]  nume\_partitie**

**Exemplu**: Vom crea un sistem de fisiere pe a doua partitie a discului **hda**:

**mkfs  -t   ex2fs   /dev/hda2**,  unde parametrul **-t** indica tipul sistemului de fisiere, iar /**dev** este un director special pentru sistemul **Linux** utilizat in operatiile de *comunicatii* cu *perifericele* calculatorului. Toate perifericele (*devices*) au cate o intrare in acest director.

**4.3. Structura fisierelor Linux**

Linux foloseste pentru organizarea fisierelor o structura arborescenta. Directorul de baza este *directorul radacina* (**root directory**). Din acest loc se construieste tot sistemul de fisiere Linux. Spre deosebire de Windows, in Linux chiar si perifericele, hard-diskurile, etc. apar ca fiind localizate in aceasta structura ierarhizata, sub directorul radacina. Structura arborescenta este urmatoarea:

**4.3. Sisteme de gestiune a fisierelor in Linux**

**4.3.1. Ext2**

**Ext2** sau **second extended filesystem** este un sistem de gestiune al fisierelor pentru sistemul de operare Linux. A fost creat de catre Rémy Card.Ext2 a fost sistemul de gestiune al fisierelor implicit in cateva distributii Linux, printre care Debian si RedHat. Desi au aparut versiuni imbunatatie, ext2 este sistemul de fisiere recomandat pentru dispozitivele de stocare de tip flash (cum ar fi carduri SD, SSD si stick-uri USB) datorita inexistentei jurnalizarii care minimizeaza numarul de scrieri, stiindu-se ca numarul de scrieri pe un dispozitiv de tip flash are un numar limitat de cicluri de scriere.

Dezvoltarea initiala a kernelului Linux a fost facuta sub sistemul de operare MINIX. Sistemul de fisiere oferit de Minix era in mare parte fara erori dar folosea un offset de 16 biti si din aceasta cauza dimensiunea limita maxima a partitiei era de 64 megabytes. De asemenea exista o limita a numelui fisierelor la 14 caractere. Din cauza acestor limite, a inceput munca pentru inlocuirea acestui sistem nativ Linux de gestiune de fisiere. Astfel a aparut sistemul de fisiere **ext.**

Sistemul de gestiune a fisierelor **ext** a rezolvat cele doua mari probleme din sistemul Minix (dimensiunea maxima a partitiei si dimensiunea maxima a numelui fisierelor la 14 caractere). Prin acesta, partitia putea avea o dimensiune de 2 gigabytes si numele fisierelor puteau ajunge pana la 255 caractere. Dar in continuare existau probleme: nu exista suport pentru acces separat, data si ora modificarii fisierelor.

Ca o solutie la caceste probleme, doua sisteme noi de gestiune a fisierelor au fost dezvoltate in Ianuarie 1993: **xiafs** si **second extended file system** (**ext2**) , care a fost o revizie generala a sistemului **extended file system** si implementa multe idei din **Berkeley Fast File System**. Ext2 a fost creat tinand cont si de o dezvoltare ulterioara cu spatii lasate in structurile sale de date de pe disk pentru utilizari ulterioare.

## 4.3.1.1. Structuri de date ext2

Spatiul este impartit in blocuri in sistemul ext2 si organizat in grupuri bloc, analog grupurilor de cilindru din UFS (Unix File System). Acest lucru este facut pentru a reduce fragmentarea externa si a minimiza numarul de cautari ale discului cand se citeste o suma mare de date consecutive.

Fiecare bloc poate contine o copie a superblockului. Superblock-ul contine informatii importante care sunt cruciale pentru bootarea sistemului de operare, astfel sunt facute copii de backup in mai multe blocuri ale sistemului de gestiune a fisierelor. Cu toate astea, de obicei numai prima copie care este gasita in primul bloc a sistemului de gestiune a fisierelor este folosita pentru bootare.

**4.3.1.2. Limitele ext2**

Motivul limitelor sistemului ext2 este formatul fisierelor si kernelul sistemului de operare. Multi din acesti factori sunt determinati odata cu crearea sistemului de gestiune a fisierelor. Acestia depind de dimensiunea blocurilor si raportul dintre numarul de blocuri si de innode-uri. In Linux dimensiunea blocului este limitat de arhitectura paginarii.

Un inode este o structura de date dintr-un sistem de gestiune a fisierelor traditional Unix, cum ar fi UFS. Un inode gazduieste informatie de baza despre un fisier, director obisnuit, sau orice alt obiect a sistemului de fisiere.

Indexarea directoarelor creste performanta pentru directoare cu un numar mare de fisiere (mai mult de 10.000). Limita teoretica a numarului de fisiere dintr-un director este de 1.3 × 1020 desi nu este relevant pentru situatii practice.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Dim. blocului | 1KB | 2 KB | 4 KB | 8 KB |
| Dim. max a fisierelor | 16GB | 256GB | 2TB | 2TB |
| Dim. max a sist. de gest a fisierelor | 4TB | 8TB | 16TB | 32TB |

**Limitari ale ext2**

## 4.3.1.3. Compresia

**e2compr** este o modificare a sistemului de gestiune a fisierelor ext2, astfel incat kernelul Linux sa suporte comprimarea si decomprimarea online a fisierelor la nivelul sistemului de gestiune a fisierelor fara niciun suport ofer de aplicatii-user.

e2compr este o adaugare, o extensie a ext2 care permite compresia si decompresia din mers. Arhiveaza doar fisiere obisnuie, datele administrative (superblockurile, inodurile, directoarele etc.) nefiind arhivate(in principal din motive de siguranta).Accesul la blocurile arhivate este posibil pentru operatii de citire si de scriere.

**e2fsprogs**

**e2fsprogs** este un set de utilitare pentru intretinerea sistemelor de gestiune a fisierelor ext2, ext3, si ext4.

Programe incluse in e2fsprogs sunt:

* e2fsck, un program [fsck](http://en.wikipedia.org/wiki/Fsck) care verifica corecteaza incosecvente
* mke2fs, folosit pentru a crea sisteme ext2, ext3 si ext4
* resize2fs, care poate mari sau micsora sistemele ext2, ext3 si ext4
* tune2fs,folosit pentru a modifica parametrii sistemelor de gestiune a fisierelor
* dumpe2fs, care afiseaza informatii a superblockului si grupurilor de blocuri
* debugfs, folosit pentru a vizualiza manual si modifica structurile interne ale sistemului de gestiune a fisierelor.

Multe din aceste utilitare se bazeaza pe biblioteca libext2fs.

**4.3.2. EXT3**

**Ext3** sau **third extended filesystem** este un sistem de fisiere jurnalizat care este folosit in mod curent de catre kernelul Linux. Este sistemul de fisiere implicit folosit de catre multe distributii populare de Linux. Stephen Tweedie a fost primul care a precizat ca lucreaza la extinderea sistemului ext2.

Marele avantaje fata de ext2 este jurnalizarea, care imbunatateste siguranta si elimina nevoia de a verifica sistemul de fisiere dupa o inchidere incorecta a sistemului.

## 4.3.2.1. Avantaje

Desti performanta (viteza) sistemului este mai putin atractiva decat la sistemele de fisiere competitoare cum ar fi JFS, ReiserFS si XFS, are un avantaj important din cauza faptului ca permite o upgradare de la ext2 fara a fi necesar backup-ul si restaurarea datelor Ext3 deasemenea foloseste mai putina putere a CPU-ului decat ReiserFS sau XFS. Este considerat de asemena mai sigur decat alte sisteme de gestiune a fisierelor pentru Linux datorita simplitatii si testarii largi a acestuia.

Sistemul ext3 adauga, fata de predecesorul sau:

* Un sistem de jurnalizare
* Creste a sistemului de fisiere online

Indexare **Hree** pentru directoare mari.

## 4.3.2.2. Dezavantaje

### 4.3.2.2.1. Functionalitatea

Deoarece ext3 îşi doreşte să fie compatibil cu ext, multe dintre structurile de pe disc sunt similare cu cele ale ext2. Ca in cazul multor sisteme de gestiune a fisierelor Linux, consistenta sistemului ext3 nu poate fi verificata in timp ce sistemul este "montat". Incercand sa verifice un sistem de fisiere care este deja montat, se pot detecta erori unde datele modificate nu au ajuns inca pe disc si poate corupe sistemul de fisiere in timp ce se incearca corectarea acestor erori.

### 4.3.2.2.2. Defragmentation

Nu exista nici o unealta de defragmentare online care sa functioneze la nivelul sistemului de fisiere. Exista totusi o unealta, e2defrag,dar aceasta cere o trecere la ext2 intai.

### 4.3.2.2.3. Recuperarea datelor

Nu exista nici un suport pentru recuperarea fisierelor sterse.Driverul ext3 sterge activ fisierele prin stergerea fisierelor inode din motive de siguranta

### 4.3.2.2.4. Compresia

Suportul pentru compresie este disponibila ca un patch neoficial pentru ext3. Acesta are nevoie insa inca de imbunatatire.

### 4.3.2.2.5. Lipsa suportului pentru "snapshot"

Fata de unele de sisteme de fisiere moderne, Ext3 nu are suport nativ pentru **snapshot** - posibilitatea de a captura rapid starea sistemului de fisiere la momente arbitrare de timp **Next3** este o versiune modificata a Ext3 care ofera suport pentru snapshots mentinand compativilitatea cu formatul EXT3.

**4.3.3. Ext4**

**Ext4** sau **fourth extended filesystem** este un sistem de gestiune a fisierelor jurnalizat pentru Linux, dezvoltat ca succesor pentru **ext3.**

## 4.3.3.1. Caracteristici

* **Sistem de fisiere mari.** Ext4 poate suporta capacitati de dimensiuni pana la 1 **exabyte** si fisiere cu dimensiuni pana la 15 terabytes.
* **Extensii.** Extensiilesunt introduse sa inlocuiasca maparea clasica cu blocuri folosita de ext2/3. O extensie este o gama de blocuri fizice continue care imbunatateste performanta fisierelor mari si reduce defragmentarea. O singura extensie ext4 poate mapa pana la 128 MB de spatiu continuu cu o dimensiune a blocurilor de 4 KB.
* **Compatibilitate**. Ext4 este compatibili cu versiunile anterioare ext2 si ext3, facand posibila "montarea" sistemelor de fisiere ext2 sau ext3 ca sisteme ext4. Acest lucru imbunateste performanta versiunilor ext2/3 deoarece le permite in acest fel sa foloseasca caracterticile noi din ext4 cum ar fi noul algoritm de alocare a blocurilor.
* **Alocare intarziata.** Ext4 foloseste o tehnica performanta a sistemelor de fisiere numita alocare intarziata. Se refera la o intarziere a alocarii blocurilor pana cand data urmeaza a fi scrisa pe disk, spre deosebire de alte sisteme de gestiune a fisierelor, care aloca blocurile necesare inainte de acest pas. Aceasta metoda creste performanta si reduce fragmentarea prin imbunatatirea alocarii blocurilor bazata pe dimensiunile fisierelor.
* **Limita de 32.000 de fisiere.** In ext3 numarul de subdirectoare pe care un director il poate contine este limitat la 32.000. Acest numar a crescut la 64.000 in ext4.
* **Verificare mai rapida a sistemului de fisiere.** In ext4, grupurile de blocuri nealocate sunt marcate. Aceasta permite lui e2fsck sa sara peste acestea la o verificare si sa reduca astfel mult timpul necesar pentru a verifica sistemul de fisiere pe care ext4 este construit sa-l suporte. Aceasta caracteristica este implementat din versiunea 2.6.24 a kernelului Linux.
* **Marcari ale datei&orei imbunatatie**. Cum computerele devin din ce in ce mai rapide si cum sistemele Linux devin din ce ince mai folosite pentru aplicatii critice, marcarea datei si orei la fisierele devine insuficienta. Pentru aceasta, ext4 extinde aceasta marcare prin masurarea in nanosecunde. Ca urmare, campul pentru aceasta marcare, a orei si datei se mareste cu 2 biti.

# 4.3.4.XFS

XFS este un sistem de fisiere pe 64 de biti. XFS este un sistem de gestiune a fisierelor cu jurnalizare de inalta performanta, creat de Silicon Graphics, pentru distributia de Linux pe care o detineau , Irix. Pentru implementare s-a folosit "Allocation Group" care se refera la divizarea discului in opt sau in mai multe zone de dimensiuni egale, fiecare din ele fiind autonoma in gestionarea spatiului, astfel incat fiecare regiune poate lucra simultan. Fata de Ext3 sau ReiserFs, XFS foloseste un sistem de jurnalizare mai performant si mai sgur prin utilizarea unei tehnici numite "Delayed Allocation" pentru alocarea mai rapida si mai inteligenta a fisierelor in spatiul liber de pe disc. (vezi si ext4).

**4.3.5.ReiserFS**

ReiserFS este un sistem de fisiere cu jornalizare dezvoltat de echipa lui Hans Reiser. Acesta lucreaza folosind metadate particulare asociate fisierelor, ceea ce îi permite sa recupereze fisierele, dupa eventualele blocaje de sistem, cu o rapiditate si o fiabilitate superioara altor sisteme.

Marele avantaj al acestui sistem de fisiere consta în faptul ca nu este legat de tehnologii anterioare precum Ext3, care este legat de Ext2. Iar unul dintre dezavantajele lui ReiserFS, fata de Ext3, este necesitatea de a face back-up si de a formata o partitie Ext2, în cazul în care se vrea conversia în ReiserFS. Momentan este în faza de dezvoltare Reiser4, succesorul lui ReiserFS,rescris aproape de la zero; printre caracteristicile sale principale se remarca o viteza majora, un suport mai bun pentru gestionarea directoarelor de dimensiuni mari continând multe fisiere, îmbunatatirea sistemului de journaling, integrarea metadatelor in interiorul spatiului numelor fisierelor, suportul pentru plug-in si optimizarea dinamica a datelor.

**4.3.6.Btrfs**

Btrfs este un sistem de fisiere pentru Linux care urmareste punerea in aplicare a unor caracteristici avansate care se concentreaza pe toleranta la erori, administrarea si reparea usoara. Initial a fost dezvoltat de Oracle, sub licenta GPL si liber pentru a putea contribui oricine isi doreste la el.

Sistemele de fisiere trebuie sa se aibe abilitatea de a adresa si administra un spatiu de stocare mare si de asemenea sa aibe aibe abilitatea de a detecta, repara si tolera erori in datele stocate pe disk.

Cele mai importante caracteristici ale sistemului Btrfs sunt:

* Utilizarea eficienta a spatiului pentru fisiere mici
* Indexare eficienta a directoarelor
* Alocare dinamica a inodurilor (explicate mai sus)
* Volume(capacitati de stocare) diferite (directoare radacina separate pentru fiecare)
* Oglinizi la nivel de obiect (mai multe copii pentru obiecte)
* Sume de verificare a corectitudinii pentru date si metadate
* Compesie

**4.3.7. SquashFS**

**SquashFS (.sfs)** este un sistem de fisiere arhivat accesibil numai pentru citire pentru Linux. SquashFS arhiveaza fisiere, inoduri si directoare si suporta blocuri de dimensiune pana la 1MB pentru o compresie mai mare.

SquashFS este folosit pe versiunile Live de pe CD-uri a distributiilor Arch Linux, Debiam Fedora, Gentoo Linux, Linux Mint, Ubuntu si pentru distributii emedded cum ar fi OpenWRT si DD-WRT pentru routeri. Este adesea folosit cu un sistem de fisiere de tip union(care permite ca mai multe sisteme de fisere sa fie montate la un moment dat) cum ar fi **UnionFS** sau **aufs** pentru a oferi capacitatea de citire-scriere pentru distributiile Live. Astfel se profita la maxim, folosind abilitatile de compresie de viteza foarte mare a SquashFS impreuna cu abilitatea de a modifica datele intr-o distributie live de pe un CD. Distributii precum Slax, Debian Live, Mandriva One si Puppy Linux folosesc aceasta combinatie.

**Bibliografie:**

http://ebooks.unibuc.ro/informatica/Seiso/4.1.htm

http://en.wikipedia.org/wiki/Ext2

http://en.wikipedia.org/wiki/Ext3

http://en.wikipedia.org/wiki/Ext4

http://en.wikipedia.org/wiki/List\_of\_file\_systems

http://en.wikipedia.org/wiki/Btrfs

http://en.wikipedia.org/wiki/Comparison\_of\_file\_systems

http://kb.siteground.com/article/What\_is\_an\_inode.html

http://ebooks.unibuc.ro/informatica/Seiso/4.1.htm

http://www.linuxcumsa.ro/Linux/structura-sistemului-de-fisiere-pe-linux/

**5. Sisteme de gestiune de fisiere in alte sisteme de operare**

**(Chirila Vlad)**

**5.1. Gestiunea fisierelor in MAC OS [1][2][7][8][9]**

Mac OS este numele unei serii de sisteme de operare bazate pe interfata grafica, dezvoltat de Apple Inc.

Primele versiuni de Mac OS erau compatibile doar cu Macintosh-uri bazate pe Motorola 68000. Cand Apple a introdus calculatoare cu hardware-ul PowerPC, sistemul de operare a fost upgradat pentru a putea rula pe această arhitectură, de asemenea. Mac OS 8.1 a fost ultima versiune care ar putea rula pe un procesor clasa 68000 (68040). Mac OS X, care a înlocuit Mac OS-ul "Clasic", este compatibil cu procesoare PowerPC şi Intel, prin versiunea 10.5 ("Leopard"). Versiunea 10.6 ("Snow Leopard") functioneaza numai pe procesoare Intel.

Macintosh folosit iniţial Macintosh File System (MFS), un sistem de gestiune al fişierelor plat cu un singur nivel de foldere. Acesta a fost rapid înlocuit în 1985 de către Hierarchical File System (HFS), care a avut un arbore director adevărat. Ambele sisteme de gestiune ale fişierelor sunt compatibile. Mai târziu, HFS a fost înlocuit cu HFS Plus sau HFS +, ca sistemul de gestiune al fişierelor primar utilizat în calculatoarele Macintosh (sau alte sisteme care rulează Mac OS). Odată cu introducerea lui OS X 10.6, Apple a renuntat la suportul pentru formatarea sau scrierea discurilor si imaginilor in HFS, care sunt suportate numai ca volume read-only.

**5.1.1. Sistemul de gestiune MFS [3][4]**

Macintosh File System (MFS) este un format de volum (sau sistem de gestiune al fişierelor de disc), creat de Apple Computer pentru stocarea fişierelor pe dischete de 400K. MFS a fost introdus cu Macintosh 128K în ianuarie 1984.  
MFS a fost remarcat, atât pentru introducerea furcilor de resurse pentru a permite stocarea de date structurate, precum şi pentru stocarea metadatelor necesare pentru suportul interfaţei grafice de Mac OS. MFS permite numele fişierelor să aiba lungimea de până la 255 de caractere, deşi Finder nu permite utilizatorilor să creeze nume mai lungi de 63 de caractere (31 caractere în versiunile ulterioare). MFS se numeşte un sistem de gestiune al fişierelor plat, deoarece nu suporta o ierarhie de directoare.

Directoare exista ca un concept pe Macintosh-urile bazate pe MFS, dar ele functioneaza cu totul altfel fata de cum o fac pe sisteme moderne. Ele erau vizibile în Windows Finder, dar nu şi în casetele de dialog. Întotdeauna exista un folder gol pe volum, şi dacă acesta era modificat în vreun fel (de exemplu, prin adăugarea sau redenumirea fişierelor), un nou folder gol ar apărea, astfel oferind o modalitate de a crea foldere noi. MFS stoca toate informaţiile despre listarea fişierelor si directoarelor într-un singur fişier. Finder a creat iluzia de dosare, prin stocarea tuturor fişierelor ca o pereche management director/management fişier. Pentru a afişa conţinutul unui anume folder, MFS va scana directorul pentru toate fişierele cu care se ocupa. Nu era nevoie sa se găsească un fişier separat care să cuprindă listarea directoarelor.

Macintosh File System nu suportă volume de peste 20 de megaocteţi, sau aproximativ 1.400 de fişiere. În timp ce acest lucru este putin după standardele de azi, părea foarte expansiv atunci, când toate computerele Apple Macintosh aveau o unitate floppy de 400 kiloocteti.

**5.1.2. Sistemele de gestiune HFS si HFS Plus [5]**

Apple a introdus Hierarchical File System ca un înlocuitor pentru MFS in septembrie 1985 pentru a oferi support pentru prima unitate hard disk Apple pentru Macintosh. În Mac OS 7.6.1, Apple a eliminate posibilitatea scrierii de volume MFS, iar în Mac OS 8.1 suport pentru volume MFS a fost eliminat cu totul.

Fişierele erau identificate cu ID-uri de fişier unice, mai degrabă decât numele de fişiere, iar numele de fisiere puteau avea 255 de caractere (deşi Finder suporta doar un maxim de 31 de caractere).

Principala preocupare a fost timpul necesar pentru a afişa conţinutul unui folder. Pentru MFS toate informaţiile de listare a directoarelor si a fişierelor erau depozitate într-un singur fişier, pe care sistemul trebuia să-l caute pentru a construi o listă a fişierelor stocate într-un anume folder. Aceasta functiona bine cu un sistem cu câteva sute de kiloocteti şi, probabil, de o sută de fişiere, dar pe masura ce sistemele au crescut la megaocteţi şi mii de fişiere, performanţa s-a degradat rapid. Soluţia a fost de a înlocui structura MFS cu una mai potrivită pentru sisteme de gestiune al fişierelor mai mari. HFS a înlocuit structura plană cu Fisierul Catalog care foloseşte o structură B-tree care ar putea fi cautata foarte repede, indiferent de dimensiune. De asemenea, HFS a reproiectat diferite structuri pentru a putea să reţină numere mai mai, numerele întregi pe 16 biti fiind înlocuite cu numere de 32 de biti aproape peste tot. Ciudat, una dintre puţinele locuri unde aceasta imbunatatire nu a avut loc a fost directorul fişierul în sine, care limitează HFS la un total de 64k fişiere.

Hierarchical File System împarte un volum în blocuri logice de 512 octeţi. Aceste blocuri logice sunt apoi grupate în blocuri de alocare, care pot să conţină unul sau mai multe blocuri logice în funcţie de mărimea totală a volumului. HFS foloseşte o valoare de 16 de biţi pentru a adresa blocuri de alocare, limitând numărul de blocuri de alocare la 65536.

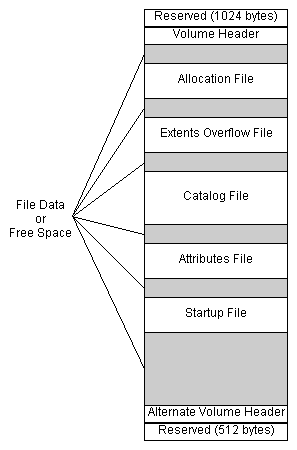
Există cinci structuri care alcătuiesc un volum HFS: blocurile logice 0 şi 1 din volum, care sunt blocuri de bootare şi conţin informaţii despre pornirea sistemului; blocul logic 2, conţinând Master Directory Block (MDB), care defineşte o gamă largă de date despre volumul în sine, de exemplu, timbrul datei si orei creării volumului; blocul logic 3, care este blocul de pornire pentru Volume Bitmap, care ţine evidenţa blocurilor de alocare care sunt în uz şi care sunt libere; fisierul de depasiri, care este un B-tree care conţine extensii suplimentare, care înregistrează care blocuri de alocare sunt alocate cărui fisier, o dată ce primele trei extensii din catalogul de fişiere sunt epuizate; şi catalogul de fişiere care este un alt B-tree, care conţine înregistrări pentru toate fişierele şi directoarele stocate în volum.

HFS Plus este arhitectural foarte asemănătoare cu HFS, deşi au existat o serie de modificări.

Obiectivele cheie care au ghidat proiectarea formatului HFS Plus au fost: utilizarea eficientă a spaţiului pe disc, folosirea de nume de fişiere compatibile international, suport viitor pentru furci de fişiere, şi uşurinţa bootarii pe sistemele de operare non-Mac OS.

HFS Plus este o versiune îmbunătăţită a HFS, sustinand fişiere mult mai mari (adresele de bloc sunt reprezentate pe 32 de biţi în loc de 16 biţi) şi folosirea Unicode pentru a denumi elementele. HFS Plus permite nume de fişiere până la 255 de caractere, şi foloseste furci multiple de fişiere similar ca la NTFS, deşi până de curând, aproape nici un software nu profita de alte furci de fisiere in afara de furcile de date si de resurse. HFS Plus foloseste, de asemenea, o tabela de alocare de 32 de biţi, faţă de HFS care folosea 12 biţi.

Structura generală a unui volum HFS Plus este ilustrată în figura de mai jos.



Există nouă structuri care alcătuiesc un volum HFS Plus tipic. Sectoarele 0 şi 1 din volum sunt blocuri de boot. Acestea sunt identice cu blocurile de boot într-un volum HFS. Sectorul 2 conţine header-ul de volum echivalent cu Master Directory Block dintr-un volum HFS. Header-ul de volum este întotdeauna situat în acelaşi loc. Fisierul de alocare care ţine evidenţa care blocuri de alocare sunt libere şi care sunt în uz. Este similar cu Volume Bitmap din HFS, fiecare bloc de alocare este reprezentat de un bit. Fisierul de alocare isi poate schimba dimensiunea şi nu trebuie să fie stocat într-un mod continuu în volum. Fişierul catalog este un B-tree care conţine înregistrări pentru toate fişierele şi directoarele stocate în volum. Fişierul catalog al HFS plus este foarte similar cu fişierul catalog al HFS. Fişierul de depăşiri este un alt B-tree care înregistrează blocurile de alocare care sunt alocate pentru fiecare fişier ca extinderi. Fişierul de atribute poate stoca trei tipuri diferite de inregistrari de 4KB: înregistrările de atribute a datelor inline, înregistrările atributelor furcilor de date şi înregistrările atributelor de extensii. Fişierul de pornire este proiectat pentru sistemele non-Mac OS, care nu au suport pentru HFS sau HFS Plus. Este similar cu blocurile de bootare ale unui volum HFS. Penultimul sector contine header-ul alternativ de volum, echivalent cu Alternate Master Directory Block al HFS-ului. Ultimul sector în volum este rezervat pentru utilizarea de către Apple. Este folosit în timpul procesului de fabricaţie al calculatorului.

**5.1.3. Avantaje ale gestiunii fisierelor in MAC OS [6]**

Majoritatea sistemelor de gestiune al fişierelor utilizate în DOS, Unix sau alte sisteme de operare tratează pur şi simplu un fişier ca o secvenţă de octeţi, care necesită o cerere pentru a şti care octeţi reprezintă ce tip de informaţii. Un fişier Macintosh conţine două componente, sau furci: furca de resurse şi furca de date. Furca de resurse este un fişier indexat ce conţine segmente de cod, elemente de meniu, casete de dialog, etc. Furca de date conţine şirul de octeţi asemanator unui fişier Unix. Fiecare furcă este formată din unul sau mai multe extensii sau şiruri discontinue de blocuri. Un descriptor de extensii codifică blocul de plecare şi lungimea extensiilor într-o cantitate pe 32 de biţi. Prima inregistrare de extensii (trei descriptori de extensii) a fiecărei furci este o parte a înregistrarilor catalogului de fişiere. Orice înregistrări ale unei extensii ulterioare sunt păstrate în B-Tree-ul de extensii. Pe langa extinderile înregistrate în fisier si B-tree un volum conţine, de asemenea, două blocuri de boot, un bloc de volum de informaţii, precum şi bitmapul spaţiului liber. Există o cantitate remarcabila de redundanţă în structurile de disc care îmbunătăţeşte recuperarea datelor dupa un crash.

**5.2. Gestiunea fisierelor in alte sisteme de operare [16]**

**5.2.1. Gestiunea fisierelor in OS/2 [10][11][12]**

OS / 2 este un sistem de operare iniţial creat de Microsoft si IBM, apoi mai târziu dezvoltat în exclusivitate de IBM. Numele vine de la "Sistem de operare / 2", deoarece a fost introdus ca parte din aceeaşi generaţie ca linia de calculatoare personale din a doua generaţie "Personal System / 2 (PS / 2)" de la IBM. HPFS sau High Performance File System este un sistem de gestiune al fişierelor creat special pentru sistemul de operare OS / 2 pentru a îmbunătăţi limitările sistemului de gestiune al fişierelor FAT.

O expunere excelentă se realizează prin utilizarea unor structuri de date avansate cum ar fi caching inteligent, citire-în-faţă şi scrie-în-spate. Spatiul de pe disc este gestionat mai economic prin folosirea sectoarelor. HPFS include, de asemenea, o toleranţă la erori îmbunătăţită substanţial. Programele au nevoie doar de simple modificări pentru a face uz de atributele extinse şi numele lungi de fişiere.  
Un volum HPFS are foarte puţine structuri fixe. Sectoarele de volum 0-15 (8KB) sunt BootBlock-ul şi conţin un nume de volum, ID-ul volumului pe 32 de biţi, precum şi un program de bootstrap al discului. Sectoarele 16 şi 17 sunt cunoscute ca SuperBlock şi respectiv SpareBlock. SuperBlock este modificată numai prin utilitati de întreţinere disc. SpareBlock conţine diverse steaguri şi indicia; acesta este modificat, deşi rar, pe măsură ce sistemul rulează. Restul de disc este împărţit în benzi de 8Mb. Fiecare bandă are propriul bitmap de spaţiu liber în care un bit reprezintă fiecare sector.

HPFS nu serveşte doar ca o modalitate de a organiza datele pe dispozitive de stocare de blocuri cu acces aleator, dar este, de asemenea, un modul software care traduce cererile orientate pe fişiere de la aplicaţii spre diverse dispozitive. HPFS este, de asemenea, un exemplu de sistem de gestiune al fişierelor instalabil, ceea ce face posibil accesul simultan la mai multe structure de volum incompatibile de pe acelasi sistem OS / 2.

**5.2.2. Gestiunea fisierelor in AmigaOS (OFS,FFS,SFS) [13][14][15]**

AmigaOS este sistemul de operare nativ al computerului personal Amiga. Acesta a fost dezvoltat iniţial de Commodore International, şi introdus în 1985, impreuna cu Amiga 1000. Primele versiuni (1.0-3.9) rulau pe seriile Motorola 68k cu microprocesoare de 16 biţi şi 32 de biţi, in timp ce noile AmigaOS 4 rulează numai pe microprocesoare PowerPC.

Primul sistem de gestiune al fişierelor Amiga a fost Old File System (iniţial cunoscut sub numele de Amiga File System). Chiar dacă a folosit blocuri de 512 octeţi, a rezervat prima porţiune mică a fiecărui bloc pentru metadate, lăsând o capacitate de bloc de date de 488 octeţi per bloc. Nu a fost foarte potrivit pentru nimic în afară de dischete, şi a fost înlocuit în scurt timp.OFS foloseste codificarea si decodificarea MFM în mod implicit atunci când opera pe dischete. Pe o dischetă Amiga există 80 cilindri. Fiecare cilindru are 2 piese MFM, unul pe fiecare parte a discului. Discurile cu densitate dubla (DD) au 11 sectoare per piesă MFM, discurile de înaltă densitate (HD) au 22 sectoare. Geometria unui disc floppy Amiga este după cum urmează: 512 bytes / sector, 11 din sectorul / piesă, 2 piese / cilindru, 80 cilindri / disc pentru discuri DD, şi: 512 bytes / sector, 22 din sectorul / piesă, 2 piese / cilindru , 80 cilindri / disc pentru discuri HD. Discul DD are 11 \* 2 \* 80 = 1760 (de la 0 la 1759) blocuri, în timp ce discul HD are 22 \* 2 \* 80 = 3520 blocuri. Amiga stochează 880 KiB pe un disc DD şi 1760 KiB pe o dischetă HD.

OFS, predecesorul lui Fast File System (FFS), folosea 24 de octeti per sector pentru redundanţa datelor, pentru a reconstrui datele structurate privind medii mai puţin sigure. Atunci când medii de stocare de viteză mai mare (de exemplu, hard disk-uri) au devenit mai disponibile pentru Amiga, aceste date redundante au reprezentat un “bottleneck”, pentru ca toate datele trebuiau să fie transferate către aplicatie pentru a fi refacute.FFS este simplu şi eficient, iar atunci când a fost introdus era mai mult decât potrivit, si avea multe avantaje în comparaţie cu sisteme de gestiune al fişierelor de alte platforme. Cu toate acestea, după cum OFS făcuse înainte, a îmbătrânit; Pe măsură ce hard disk-urile au devenit mai mari şi numărul de fişiere de pe le-au crescut, utilizarea sa a devenit tot mai problematică.  
FFS stocheazăun bitmap al sistemului de fişiere într-un singur sector, ceea ce predispune la corupţie sau ştergerea dacă o operaţie de scriere este întreruptă de un crash sau de scoaterea discului. Acest lucru ar putea duce la o situaţie cunoscută sub numele de invalidare a sistemului de fişiere. Pentru a repara aceasta, întregul disc trebuia să fie scanat şi bitmap-ul reconstruit. AmigaOS includea un validator de disc, iniţial pe disc (care a fost, din păcate, predispus de a fi inlocuit de virusi pentru a se răspândi) sau, ulterior, ca parte a ROM. Validatorul de disc încerca să repare bitmap-ul prin protejarea impotriva scrierii a discului şi scanarea lui; acest lucru putea dura o lungă perioadă de timp şi făcea accesul la disc foarte lent până când termina, mai ales în cazul mediilor mai lente.  
Deşi iniţial limitată la adresare pe 32 de biţi (şi, prin urmare, discuri de aproximativ 2 GB), FFS a primit ulterior unele patch-uri pe 64 de biţi ale unor părţi terţe şi apoi actualizări reale care să îi permită să ocolească aceste limitări. Cele mai multe dintre aceste instrumente nu au fost actualizate la momentul în care FFS devenit capabil de adresarea pe 64 de biţi şi putea să funcţioneze pe partiţii doar cu primii 2 GB de pe disc.Din aceste motive, FFS a fost înlocuit la mijlocul anilor 1990 cu alternative mai la zi, cum ar fi Smart File System (SFS).Smart File System (SFS) este un sistem de gestiune al fişierelor care ţine evidenţa modificărilor pe care intenţionează să le facă într-un jurnal (de obicei, un jurnal circular într-o zonă dedicată a sistemului de fişiere), înainte de comiterea lor la sistemul de fişier principal. În cazul unui crash de sistem sau pană de curent, astfel de sisteme de gestiune al fişierelor sunt mai rapide pentru recuperarea datelor şi mai puţin susceptibile să devină corupte. SFS utilizează dimensiuni ale blocurilor variind de la 512 (29) la 32768 (215) octeti cu o dimensiune maximă a partiţiei de 128 GB. Integritatea este menţinută prin ţinerea unui jurnal de rollback cu toate modificările asupra metadatelor într-o anumită perioadă de timp. Jurnalul este scris pe disc întai în spaţiul liber şi apoi blocurile de metadate sunt suprascrise direct. Spre deosebire de celelalte sisteme de gestiune al fişierelor Amiga, FFS şi OFS, integritatea sistemului de fişiere este foarte rar compromisă. O caracteristică deosebit de interesant a SFS (şi aproape unică pentru sisteme de gestiune al fişierelor Amiga), este capacitatea sa de a se defragmenta în timp ce sistemul de fişiere este folosit, chiar şi pentru fişierele blocate. Procesului de defragmentare este aproape complet fără stare (independent de locatie), ceea ce înseamnă că poate fi oprit şi a început instantaneu. În timpul defragmentării integritatea datelor este asigurată atât de metadate cât si de date normale.

**Bibliografie:**

[1]http://developer.apple.com/mac/library/documentation/MacOSX/Conceptual/BPFileSystem/BPFiBPFileSy.html

[2] http://www.asy.com/files.htm

[3] http://mac.wikia.com/wiki/Macintosh\_File\_System

[4] http://en.wikipedia.org/wiki/Macintosh\_File\_System

[5] http://developer.apple.com/mac/library/technotes/tn/tn1150.html

[6] http://tldp.org/HOWTO/Filesystems-HOWTO-7.html

[7] http://arstechnica.com/hardware/news/2008/03/past-present-future-file-systems.ars/4

[8] http://disktype.sourceforge.net/doc/ch03s02.html

[9] http://osxbook.com/book/bonus/ancient/whatismacosx/arch\_fs.html

[10] http://tldp.org/HOWTO/Filesystems-HOWTO-4.html

[11] http://hobbes.nmsu.edu/h-browse.php?dir=/pub/os2/info

[12] Microsoft Systems Journal Sept 1989

[13] http://en.wikipedia.org/wiki/Amiga\_Old\_File\_System

[14] http://en.wikipedia.org/wiki/Amiga\_Fast\_File\_System

[15] http://en.wikipedia.org/wiki/Smart\_File\_System

[16] http://lclevy.free.fr/adflib/adf\_info.html