## Introducere in Unix

- Introducere
- Caracteristici
- Istoric

## Introducere in Unix

- Introducere
  - Familiarizarea cu:
    - Sistemul de operare UNIX
    - Alte resurse
  - Studiul unor aspecte:
    - Administrare
    - Aplicații distribuite
- Sistem Unix V Release 4
  - Caracteristici
  - Principii
  - Istoric

## Introducere in Unix

- **Dezvoltat în anii 70 la:**
  - **Compania AT&T**
  - **Universitatea Berkley**
- **Sistem de operare**
  - **Industrie**
  - **Cercetare**
  - **Administrație**
  - **Educație**
- **Lucrul în rețea, servere de rețea**

## Introducere in Unix

- **Variante:**
  - **System V, BSD, XENIX, AT&T, AIX, Linux, UINIXWare, Solaris, IRIX, s.a.**

- **Nu s-a impus un Standard unic**
  - **Diferențe de implementare şi exploatare**
  - **Aceleaşi principii utilizate**

- **UNIX - Denumirea generică a unei largi familii de sisteme de operare:**
  - **Portabile**
  - **Orientate pe comenzi**
  - **Multiuser**
  - **Multitasking**

## Introducere in Unix

- ***Sistem de operare -*** SO: sistem de programe de bază care realizează controlul şi managementul resurselor calculatorului, oferind şi un set de servicii utilizatorului. Funcţii realizate de SO
  - Planificarea, încărcarea, iniţializarea şi supervizarea execuţiei programelor
  - Locarea memoriei
  - Iniţializarea şi controlul operţiilor de I/E
  - Tratarea erorilor

- ***SO portabil:*** poate fi transferat uşor de la un sistem de operare la altul

- ***SO orientat pe comenzi*****:** sistemul dispune de un interpretor de comenzi care preia c-zile introduse de utilizator, le execută şi afişează rezultatele obţinute
- ***SO multiutilizator***: sistemul permite crearea de ***conturi utilizator***; utilizatorii se pot conecta şi lucra simultan, au anumite drepturi şi restricţii de acces la fişiere şi la celelalte resurse ale sistemului, impuse prin mecanisme de protecţie

- ***SO multitasking*****:** permite execuţia simultană a mai multor programe, numite procese (taskuri). Multitaskingul se realizează prin mecanismul de time-sharing în cazul calculatoarelor monoprocesor. CPU alocată proceselor conform unei politici de planificare:
  - Round-robin
  - Bazată pe priorităţi statice sau dinamice

## Principii

- Modularitate
- Operaţiile de I/E + integrate în sistemul de fişiere realizându-se: ***I/E generalizate***
- Există un sistem de gestiune a proceselor reentrante şi asincrone multiple, care se pot sincroniza prin intermediul unui sistem de întreruperi logice
- Gestiunea memoriei: printr- un  mecanism care permite schimbul de pagini între  memoria RAM şi cea externă
- Interfaţa simplă prin componenta shell, care nu este integrată în kernel
- Sistem deschis – componente portabile, scrise in C
- Întreţinere şi dezvoltare simple

## Istoricul sistemului de operare Unix

- UNIX System V
  - Bell Laboratories 1969: Ken Thomson, dezvoltă sistemul Unics (Uniplexed Informaion and Computing System) pe sisteme minicalculator DEC PDP-7. scris în limbaj de asamblare Fortran
  - Prima documentație Unix este scrisă în 1971
  - În 1972 Limbajul C (Dennis Ritchie – Bell Laboratories) – Unix rescris în C de Ritchie şi Thomson, devenind multitasking.
  - 1973 + apare prima versiune poratabilă distribuită gratuit univ, şi sub licență guvernului SUA şi firmelor
  - 1974 – "The Unix Time-Sharing System" – "***a small and yet powerfull operating system"***
  - 1975 versiunile 4, 5, 6 care definesc *pipes*
  - 1978 ultima versiune de cercetare: 7, pe DEC PDP-11, nucleu independent de hardware – **prima versiune comercializată**
  - 1982 Unix System III – VAX 11/780
  - 1983 Unix System V
  - 1980-1981 primele licențe: ULTRIX (DEC), XENIX (Microsoft), UTS (Amdahl) etc.

## BSD Unix

- Versiunea 7
  - **Dezvoltări realizate de AT&T(Bell Lab)** – versiuni succesive System V Unix
    - Semafoare, blocaje, cozi de mesaje, memorie virtuală, memorie pe 8 biți
  - **Dezvoltări realizate de Universitatea Berkeley** – BSD Unix ( Berkeley Software Distribution) -Thomson(1976)
    - Memoria virtuală (BSD 4.1)
    - Facilități de rețea TCP/IP (BSD 4.2)
    - Schimb de info între procese centralizat sau distribuit

- Alte variante Unix (plecând de la nucleul AT&T)
  - XENIX – Microsoft
  - VENIX – Venturecom
  - **AIX – IBM**
- Sistem ne – AT&T
  - MINIX – Andrew Tanenbaum - Amsterdam
  - Linux – Linus Torvald -1991(RedHat, SuSe,...)
  - XINU – Douglas Corner
  - GNU – FSF (Free Software Fundation)

- Portarea aplicațiilor dificilă → definirea de interfețe standard:
  - X/OPEN şi POSIX ( Standard IEEE Portable Operating System Interface for Computer Environments) au preluat propunerile făcute în SVID( System V Interface Definition)
  - Normalizarea sistemului doar la nivel utilizator:
    - Unix International (AT&T şi Sun)
    - OSF ( Open Software Foundation)

- Frână pt răspândirea Unix – aspectul neprietenos al interfeței utilizator – bazată pe: utilizarea de terminale alfa numerice ( mod text nu grafice)
- Adoptarea protocolului X – Windows – interfața utilizator grafică şi dezvoltarea de medii grafice bazate pe acest protocol
- **Sistemul X** (proiectul Athena MIT)
  - Distribuirea calculelor între diferite unități centrale, stații de lucru a utilizatorilor şi celelalte maşini din rețea( ale utilizatorilor)
  - Adoptat ca standard: biblioteci grafice
- Protocolul NSF(Network File System), Sun System, sistem distribuit fizic

## Structura SO Unix

- UNIX – sistem de operare scris pentru programatori şi utilizatori
- Structura unui sistem Unix
  - Nucleu
  - Shell
  - Comenzi şi aplicații utilizator
- Nucleul (kernelul) – partea de bază, rezidentă din SO care interacționează direct cu hardware-ul, prin intermediul unor drivere pentru dispozitivele fizice.
- Funcțiie  princpale ale nucleului sunt:
  - Gestionarea memoriei
  - Gestionarea / planificarea accesului la resurse
  - Gestionarea sistemului de fişiere
  - Tratarea întreruperilor
  - Tratarea erorilor
  - Gestionarea operațiilor de I/E de nivel scăzut
- C-zile şi aplicațiile interacționează cu nucleu prin apeluri sistem(system calls). Există definite aproximativ 100 apeluri sistem care realizează deschiderea, citirea, scrierea, închiderea fişierelor, execuția, terminarea unui poces, schimbarea priorității unui proces,...
- Programele care realizează apeluri sistem sunt  scrise in C
- Cea mai mare parte a nu cleului este scrisă în C.

## Structura SO Unix

- Shell = interfața între utilizator şi kernel = interpretor de comenzi
  - Citeşte
  - Interpretează
  - Execută comenzile
- Tipuri:
  - C Shell
  - Bourne Shell
  - Korn Shell
- Shellul este în acelaşi timp şi un limbaj de programare întrucât permite scrierea de programe ( sau scripturi) shell
- C-zile Unix se înscriu în categoriile:
  - De gestiune a proceselor
  - De manipulare a fişierelor
  - Editoare de text
  - De comunicare între utilizatori
  - Compilatoare, depanatoare
  - Programe pentru poşta electronică şi alte protocoale de acces Internet– componente portabile, scrise in C
- Aplicații complexe distribuite, Web, multimedia, pot fi realizate pornind de la utilitarele, c-zile definite de SO

## Sistemul de fisiere UNIX

Fişierele sunt parte integrantă în Unix. Un fişier este cea mai mică unitate în care informația se memorează, o secvență de octeți servind unui scop comun.

Sistemul de fişiere se referă la activitățile de:

***- structurare***
***- denumire***
***- accesare***
***- utilizare***
***- protecție***
***- implementare***

În Unix fişierele pot fi:

- ***fizice***

- ***virtuale***: pt a realiza coexistența mai multor sisteme de fişiere diferite, cu rolul de a uniformiza accesul la fişiere

- ***distribuite***: permite accesarea de fişiere stocate la distanță, pe calculatoare din rețea

Fişierelor au o organizare ierarhizată, pe directoare(cataloage) formate la rândul lor din fişiere şi subdirectoare.

**Partiția**

- Sistemele fizice sunt plasate înt-o zonă a unui mediu de stocare (disc magnetic) numită ***partiție***.

- Disc – mai multe partiții pe fiecare contruindu-se câte un sistem de fişiere independent. Fiecare partiție se comportă, la nivel utilizator, ca un disc de sine stătător.

- Pentru a putea fi utilizat discul trebuie formatat, fiind împărțit în blocuri

- Mărimea unei partiții – stabilită de către administratorul de sistem la instalarea sistemului sau când se introduce un nou disc în sistem.

- Fiecare partție e gestionată în mod separat de sistemul de operare ca şi cum ar fi un dispozitiv separat.

**Partiția**

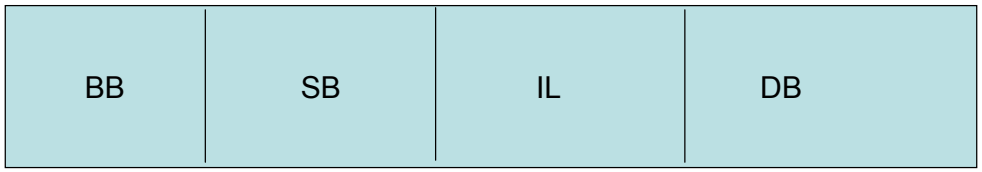
Gestiunea fişierelor dintr-o partiție - informații despre:

- blocurile de date libere/ocupate
- fiecare fişier din partiție

În DOS partițiile sunt discuri logice: C:, D:, E:, s.a.m.d.

În Unix partițiile sunt specificate prin fişiere speciale aflate în directorul **/dev** , care pentru folosire trebuiesc montate.

Sistemul de fişiere foloseşte ca unitate de bază ***blocul de pe partiție*** – orice alocare se face la nivel de bloc. Mărimea unui bloc este multiplu de 512 octeți, depinzând de implementare sistemului.




**Modul de organizare a blocurilor**

***BB - blocul de boot*** - conține programele care realizează încărcarea părții rezidente a sistemului de operare Unix; ocupă de obicei primul sector al discului

***SB – super blocul*** – conține info despre întreaga partiție – modul cum sistemul de fişiere foloseşte partiția:

- mărimea şi starea sistemului de fişiere, eticheta, numele sistemului de fişiere, mărimea blocurilor, data şi ora ultimei modificări a superblocului
- info despre i-noduri: număr de i-noduri alocate/libere
- blocurile de memorie

***IL – lista de i-noduri*** cuprinde info cu privire la fiecare i-nod din sistem; unul dintre i-noduri este rădăcinaa sistemului de fişiere prin acesta fiind disponibilă structura de cataloage a sistemului de fişiere (i-node = infoemation node)

***DB – blocuri de date*** care conțin info efective ale directoarelor de fişiere

***swap*** – mai există şi o zonă de swap, rezervată pentru păstrarea imaginilor proceselor atunci când sunt eliminate temporar din memorie pentru a face loc altor procese; de obicei pentru zona swap se folosesc partiții distincte

**Sistemul de fisiere UNIX**

Categorii de obiecte:

- ***fisiere ordinare:*** sunt folosite pentru memorare de informații pe suport magnetic. Pot fi text sau binare. Intern fişierele nu au nume ci sunt identificate de un număr numit i-number care reprezintă indexul unui şir de i-noduri

- fişiere ***cataloage***: sunt tot ***fisiere*** dar structura lor interna e ***cunoscuta si gestionata de sistemul de operare , folosita pentru a realiza structura ierarhica a sist de fisiere***

***- fişiere speciale:*** *fişiere asociate* ***dispozitivelor fizice*** *(discuri, imprimante, mouse) sau* ***virtuale*** *(memoria internă, terminale). Utilizatorii pot efectua schimburi de date cu perifericele fără a fi nevoişi să cunoască detalii despre mecanismul de funcşionare a acestora, tratarea lor fiind uniformă cu a fişierelor obişnuite şi beneficiind la fel de mecanismul de protecție*

**Sistemul de fisiere UNIX**

**Numele unui fisier sau catalog:**

- Se poate compune cu toate caracterele acceptate de UNIX cu excepția */* care e delimitatorul de nume director. Se face distincție înter literele mari şi mici – case sensitive
- Se evita **caracterele care au o semnificație specială în shell**

**! # & @ $ ^ ( ) ‘ “ ; | < > [ ] * ? **

- Limita lungimii numelui unui fisier: **255 caractere**;
- Nu există conceptul de **extensie** a numelui: se utilizeaza ultimele caractere din nume incepind cu ”.” pt a da indicatii asupra **naturii informatiilor** din fisier:
  - .c - fisierul contine **text sursa C**;
  - .so - fisierul contine o **biblioteca partajabila**;
  - .sh - fisierul este o procedura shell;
- ***Un fişier poate avea mai multe nume dar există un singur i-nod asociat. Relația stabiltă între nume şi i-nod se numeşte legătură.***

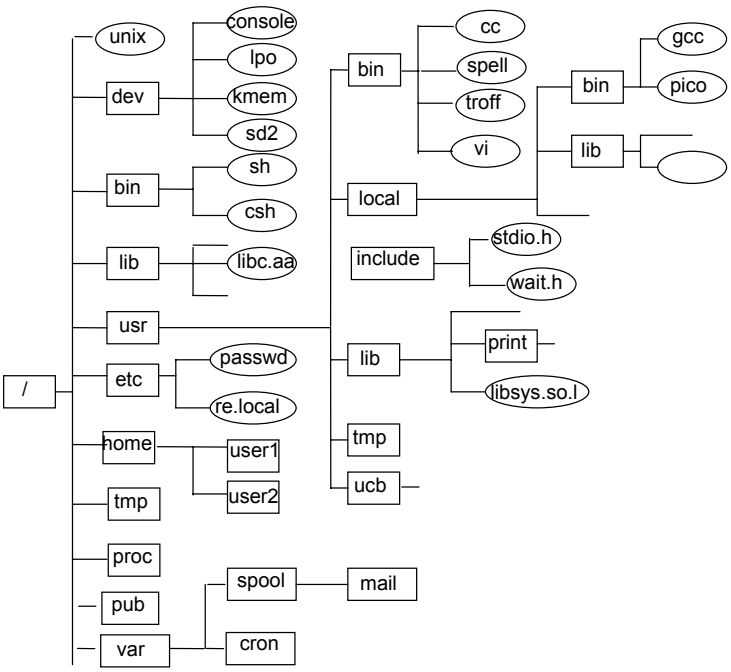
- ***Fişiere director***
  - Directoarele sau cataloagele oferă posibilitatea de accesare a fişierelor prin nume, nu prin i-nod, deasemenea modalitatea de structurare ierarhică a acestora.
  - Directoarele = fişiere care conțin info despre subdirectoarele şi fişierele conținute
  - Directorul root, notat */***,** este rădăcina structurii arborescente.
  - Identificarea unui fişier se poate face precizând calea (path) şi numele. Calea se poate preciza absolut, pornind de la root, sau relativ la directorul curent
  - În cadrul fiecărui director există două fişiere numite **.** şi **..** care semnifică directorul curent şi pe cel rădăcină

**Legături**

- Relația stabilită între nume şi i -nod = legătură
- Legătura = un alt nume (alias) al fişierului
- 2 utilizatori lucrează asupra aceluiaşi document – nu se păstrează câte o copie a doc - soluția: utilizarea unei legături, utilizatorii partajând acelaşi fişier chiar dacă acesta apare utilizatorilor cu nume diferite
- Tipuri de legături:
  - Simbolice – symbolic links- fişierul legătură conține numele fişierului original
  - Fizice – hard links
- Linkurile permit asignarea mai multor nume unui fişier ordinar, dar nu şi unui director. Nu se pot crea legături decât între nume de fişiere situate pe acelaşi computer. Când se operează asupra unui fişier legat, se lucrează de fapt asupra fişierului de bază pe care legătura îl indică.

Structura generală de directoare

- Structura generală a arborelui directoarelor etse standardizată, fiind tipică pentru toate sistemele Unix



**Sistemul de fisiere UNIX**

***Sistemul de fisiere- caracteristici***:

- realizat sub forma ***unui singur arbore*** (***radacina unica***);

- suportul fizic pe care se gaseste acest **catalog radacina** nu e vizibil programatorului/utilizatorului;

- ***numele de cale al unui fisier*** se compune din numele tuturor cataloagelor care trebuie parcurse de la catalogul radacina inclusiv, pt a ajunge la fisierul in cauza plus numele fisierului in catalogul ultim:

**/usr/local/bin/pico**

- ***nume de cale absolut***: incepe cu / (incepe de la catalogul radacina)

- ***nume de cale relative***: nu incep de la catalogul radacina; De expl **bin/pico**; Se foloseste implicit o info pastrata de sist de operare sub numele ***catalog de lucru*** care se ataseaza in fata numelui de cale relativ→nume de cale absolut .

- **Catalogul radacina** contine numai nume de cataloage cu exceptia unui fisier care reprezinta imaginea nucleului sist de operare;

- **Catalogul/dev** contine fisiere care corespund **perifericelor**: terminale, imprimante, discuri, **linii de comunicatie**, mouse, dar si **periferice virtuale**;

- **Catalogul /bin** este destinat sa contina sub forma de ***fisiere*** in format ***executabil*** principalele **comenzi ale sistemului** ( vezi interpretorul de c-zi **sh**). Poate sa apara doar ca o leg simbolica spre alt catalog **/usr/bin**.
- **Catalogul /lib** contine o serie de biblioteci (***colectii de functii***) ale sistemului, fie sub forma de arhiva( nume de ***fisier cu sufixul .a***), fie ca si cod obiect partajabil (***fisiere cu sufixul .so***).
- **Catalogul /etc** contine fisiere si cataloage destinate mai ales administrarii sistemului: **informatii de configurare** si o serie de **c-zi speciale**.

  **Atentie: /etc/passwd** pastreaza info despre utilizatorii sistemului: numele, parola de acces ( criptata), catalogul gazda, interpretorul de c-zi implicit,..

  **/etc/rc.local** reprezinta o procedura shell (“shell script”) care se executa la incarcarea sistemului si efectueaza initializarile specifice oricarui calculator

- **Catalogul /home** contine cataloagele gazda ale utilizatorilor. Pt fiecare utilizator inregistrat in sistem se creaza un catalog in care se ajunge de fiecare data cind utiliz incepe o sesiune de lucru (info inregistrata in **/etc/passwd** ). Aici utilizatorul isi organizeaza fisierele cum doreste.

- **Catalogul /tmp** – pt a crea ***fisiere temporare*** de catre diverse c-zi ale sistemului cit si de diversi utilizatori. Trebuie sterse in momentul in care nu mai sunt necesare.
- **Catalogul /proc** - are intrari care corespund proceselor active - este un **pseudo sistem de fisiere**. Acces la info de stare ale proceselor.
- **Catalogul /pub** – nu exista in toate sistemele. La calculatoarele cu ***rol de server***, in special de ***server de fisiere***, in acest catalog incepe ***ierarhia*** sistemului de fisiere care se pun la disp utilizatorilor.
- **Catalogul /var** – contine o serie de subcataloage destinate unor functii de interes general in sistem - vezi:
  - **/var/spool** unde se gaseste ***subcatalogul pt cutiile postale ale utilizatorilor.***
  - **/var/log** destinat pastrarii diverselor **fisiere jurnal** ( evenimente petrecute in sistem: **intrari si iesiri din sesiune, erori,..)**
- **Catalogul /mnt – *montare temporara*** a unor sisteme de fisiere care devin accesibile in cadrul ierarhiei UNIX.
- **Catalogul /opt** pt a instala ***module optionale*** din distributia unui sist UNIX si chiar aplicatii.

- **Catalogul usr** este inceputul unei ierarhii complexe; particularizeaza **config.sist de fisiere** int-un anumit calculator.
  - In **/usr/bin** apar majoritatea c-zilor sistemului- ordinul sutelor
  - Catalogul **/usr/include** contine fisiere antet utilizate de aplicatiile sistemului sau necesare comenzilor(vezi **stdlib.h**- pt programe C).
  - Catalogul **/usr/lib** ca si **/lib** contine **fisiere biblioteca**.
  - O serie de c-zi apar in **/usr/ucb**
    - **du** - pt aflarea spatiului ocupat pe disc de un catalog;
    - **df** – pt aflarea spatiului liber pe un disc;
- **Catalogul /usr/local** colecteaza acele comenzi aplicatii si informatii care particularizeaza efectiv un anumit calculator in raport celelalte . Daca utiliz introduce o aplicatie noua aceasta e depusa in /**usr/local**.
- Multe dintre c-zile destinate ***administrarii sistemului*** se gasesc in cataloage distincte nereprez in fig. **/sbin si /usr/sbin**.
- /**usr/X (/ usr/ X11R6** la LINUX **)** este inceputul ierarhiei programelor pt interfata grafica
- /**usr.src** contine sursele unor fisiere antet utilizate in programarea de sistem
- /**usr/man** paginile de manual de referinta **on-line** pt: c-zile sist., apelurile sist., functiile de biblioteca, formatul fisierelor de configurare etc.

## Montare fişiere

- Fişierele speciale care indică unități de disc sau partiții sunt folosite în operația numită **montare** a sistemelor fişiere.
- Sistemul Unix/Linux permite montarea într-un director a unui sistem de fişiere aflat pe un disc sau o partiție
- ***După montare, în directorul respectiv se va afla întreaga structură de fişiere şi directoare de pe sistemul de fişiere respectiv.***
- Dacă se adaugă şi sistemul de fişiere NFS ( Network File System), structura de directoare va conține şi sisteme de fişiere montate de la distanță ( de pe altă maşină)

## Organizarea spatiului pe discuri UNIX

**Partitionarea(spatiul de pe disc se imparte)** ← ***programe utilitare***

**Partitie: cuprinde pistele cu acelasi nr de ordine de pe toate fetele utile ale unitatii de discuri.**

**Cerinta:** sa existe cel putin doua partitii:

**-** una pt **sistemul de fisiere uzual;**

**-** alta ca **suport pt memoria virtuala( partitia de swapping)- daca lipseste apar degradari de performanta;**

**Din considerente practice se lucreaza cu mai multe partitii:**

**- subsistemul /home** apare ca o partitie separata asigura protectia info utilizatorilor

- pe acelasi disc partitii care apartin unor sisteme de operare diferite.

**OBS nr. de partitii, denumirea acestora, si detaliile de divizare si formatare difera de la o distributie UNIX la alta.**

**Fiecare partitie UNIX ( nu swapping) este suportul unui sistem de fisiere distinct, care apare a avea propriul catalog radacina)**

**Divizarea spatiului unei partitii se face in urmatoarele zone:**

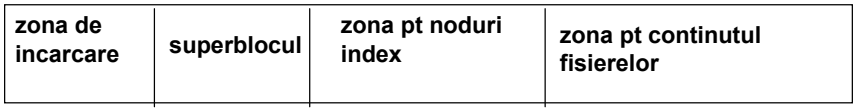
**-zona de incarcare;**

**- superblocul;**

**- zona nodurilor index;**

**- zona pt continutul fisierelor**

- ***Zona de incarcare*** – contine un program pt incarcarea nucleului sistemului si activarea acestuia, daca partitia e destinata sa devina ***radacina a sistemului de fisiere***
- ***Superblocul*** descrie ***starea sistemului de fisiere***: dimensiunea, nr. de fisiere care pot fi create, unde se gaseste spatiu liber pe disc, etc.
- ***Zona nodurilor index*** - pastreaza info despre fiecare fisier creat in partitie. Nodurile index apar in aceasta zona ca o lista liniara (tablou), in care un nod se identifica printr-un indice. Se rezerva cite un nod index fiecarui fisier fizic.
- ***Zona*** pt continutul fisierelor- se gaseste imediat in continuarea zonei nodurilor index si ocupa tot restul spatiului partitiei. In UNIX ***alocarea spatiului*** pt un fisier se face la ***cerere***, pe masura ce fisierul creste in dimensiune folosindu-se ca unitate de alocare un bloc.



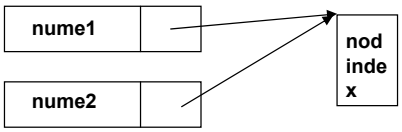

**Structura unui nod index**

Nod index sau **i-nod** pastreaza ***inform esentiale*** despre un fisier si trebuie vazut ca o ***extensie logica*** a unei ***intrari de catalog***.

Intrarile de catalog in Unix:

**nume fisier+ nr nodului index**

***Legatura intre intrarile de catalog si nodurile index:***




Cele 2 intrari de catalog se redera la acelasi nod index deci la acelasi fisier fizic

Informatia dintr-un i-nod e organizata in cimpuri.

Informatia din /etc/passwd referitoare la un utilizator:

**ada:aTx2875x9GMFE: 200:100:Ada Popa: /home/ada: /bin/bash**

Sau uneori

**ada: x: 200:100:Ada Popa : /home/ada:/bin/bash**

**1 cimp**: **ada** – numele de login

**2 cimp**: **aTx2875x9GMFE** (parola criptata) sau **x-** parola utilizatorului( parola criptata e in fisier ascuns)

**3 cimp**: numeric- identificatorul utilizatorului: **UID**

**4 cimp**: numeric- identificatorul de grup al utilizatorului: **GI**

**5 cimp**: cimp de comentariu- numele complet al utilizatorului

**6 cimp** catalogul gazda al utilizatorului

**7 cimp** : numele cale al interpretorului de comenzi pt acest utilizator

**Prezentarea cimpurilor dintr-un i-nod**

1) **Numarul utilizatorului:** este **UID-ul din /etc/passwd-** preciz. proprietarul, influenteaza drepturile de acces la fisier. Este stabilit de sistem la crearea fisierului.
2) **Grupul proprietarului: este GID-ul din /etc/passwd-** drept de acces
3) **Bitii de protectie: 3 tipuri de utilizatori:**

   **- owner: proprietarul fisierului;**

   **- group: utilizat care fac parte din acelasi grup ca si proprietarul**

   **- world: toti ceilalti utilizatori care pot obtine acces la fisiere;**

**Pt fiecare categorie de utilizatori se introduc: dreptul (sau absenta) de:**

**-citire a continutului fisierului;**

**- scriere a continutului fisierului( daca e executabil);**

**- lansarea in executie;**

4) **timpii celor mai recente operatii**; se pastreaza distinct:

-timpul ultimului acces la fisier;

-timpul ultimei actualizari (modificari);

- timpul ultimului acces pt actualizarea noului index;

5) **codul fisierului** – informatii asupra naturii acestuia ( obisnuit, catalog, fisier care reprez un periferic,…)

6) **contorul de legaturi -** in cite intrari de catalog e descris acest fisier fizic ( imp. la stergerea fisierului)

7) **lungimea curenta a fisierului**( in oteti) – poate diferi de dim. spatiului alocat pt ca alocarea se face la nivel de bloc;

8) **lista de blocuri alocate fisierului** - ***tabel de dimensiune fixa.***

***Blocul= unitatea de alocare a spatiului in UNIX = un anumit nr de sectoare de disc.***

sectorul= **512** octeti

bloc= **2-16** sectoare → **1-8** kocteti

Fiecare intrare a tabelei contine cite o adresa de bloc ( pe 4 octeti), dar blocurile adresate au naturi diferite.

Pentru a indica ce blocuri compun un fişier se folosesc pointeri la blocuri.

***pointer la un bloc*** = numărul de pe partiție al blocului

Blocul 0 (bloc de boot) nu aparține nici unui fişier

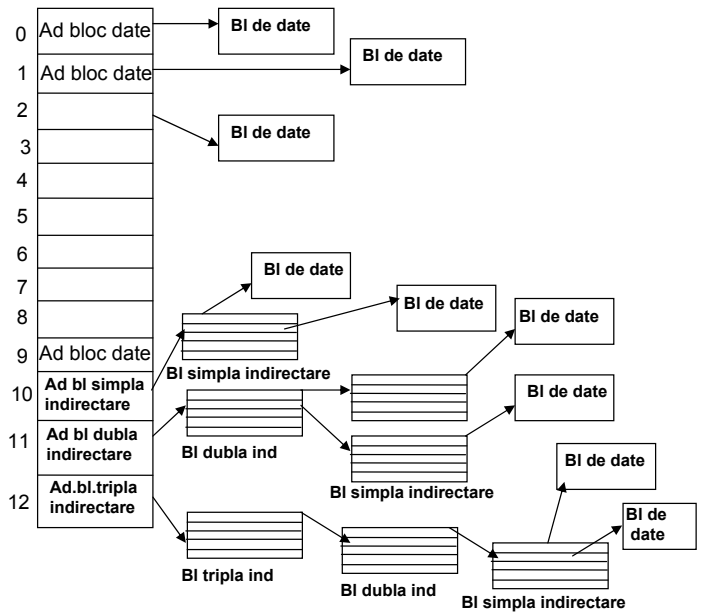
Modul în care info din i-node indică blocurile ce fac parte din fişier:

- ***pointeri spre primele 10 blocuri ale fişierului***

- dacă fişierul e mai lung de 10 blocuri, apare un pointer spre un bloc, ***blocul simplu indirect***, în care se pun pointerii spre următoarele blocuri

- dacă fişierul are mai multe blocuri decât se pot reprezenta astfel, urmează un pointer la un bloc, în care sunt pointeri la blocuri care țin pointeri la blocurile fişierului: ***indirectare dublă***

- dacă nu e suficient se realizează o ***indirectare triplă***




1-9 adreseaza direct bl de date: **dimens bl=1koctet →blocurile de date ale fisierelor mai mici de 10 kocteti se pot adresa de aici.**

**1 intrare- adresa unui bloc de simpla indirectare- nu contine date ci adrese de blocuri de date → 256 adrese bl date**

**1 intrare-adresa unui bloc de dubla indirectare- adrese ale unor blocuri simpla indirectare**

**1 intrare- bloc tripla indirectare- adresa unui bloc tripla indirectare- adrese ale unor blocuri dubla indirectare**

**Spatiul alocat fisierelor mari in zona pt continutul fisierelor consta din:**

**- blocuri care contin date pr-zise;**

**- blocuri care contin info de evidenta a spatiului alocat fisierului**

**Dezavantajele indirectarii:**

**- Degradarea performantelor (accese repetate pt a ajunge la un bloc de date)**

**Solutii: zone tampon in memoria principala pt a pastra inf din fisisere ( aceasta poate diferi pe anumite intervale de cea de pe disc, se rescrie din 30 in 30 sec).**

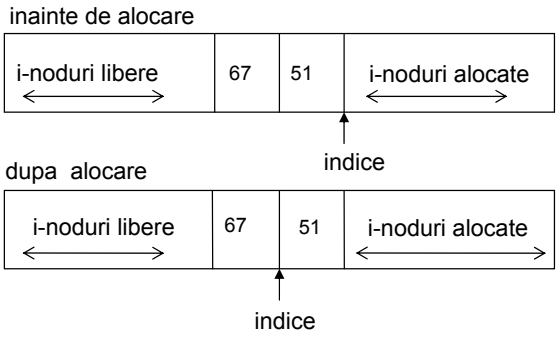
**Alocarea nodurilor index**

**Info pastrate in superbloc**:

- nr de i-noduri libere din sistemul de fisiere;
- o lista de i-noduri;
- indicele urmatorului i-nod liber din lista.

Cind se creaza un fisier nou trebuie sa i se aloce un **i-nod din cele libere.**

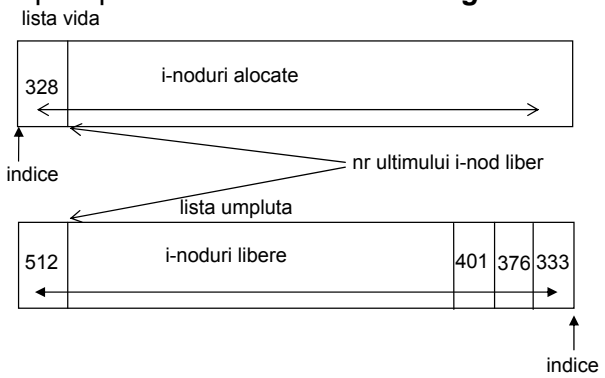
***a)*** ***Lista are i-noduri libere*** - se extrage nodul spre care arata indicele(51) si se actualizeaza val indicelui.




b) ***lista de i-noduri libere din superbloc e vida*** . Incepe cautare unor noduri libere (prezenta unui 0 in cimpul care indica tipul fisierului) pe disc si inregistrarea lor in lista din superbloc pina la umplerea acesteia. Cautarea incepe de la nr maxim de i-nod pastrat in sistem. Si va conduce la o noua valoare a acestuia cind lista e plina.

Situatia din fig: - dupa umplerea listei dar inainte de a aloca un nod index( 333)

Obs: cind se elibereaza un i-nod( la stergerea unui fisier) in i-nodul respectiv se inscrie zero in cimpul tipului fisierului → ***Acesta e gasit la o cautare pe disc.***



## Alocarea blocurilor

Info pastrate in superbloc:

- nr de blocuri libere;
- o lista de blocuri libere;
- indicele urmatorului bloc liber din lista.

La crearea sistemului de fisiere intr-o partitie( prin **mkfs**) blocurile din zona pt continutul fisierelor se introduc intr-o lista inlantuita. Fiecare element al listei e un bloc care ***contine adresele unor blocuri libere pe disc*** .

Ultima reprezinta adresa urmatorului bloc care face parte din lista.

Blocul de la inceputul listei e accesibil in superbloc.

Cind nucleul sistemului de operare trebuie sa aloce un nou bloc , va lua urmatorul bloc din lista continuta in superbloc.

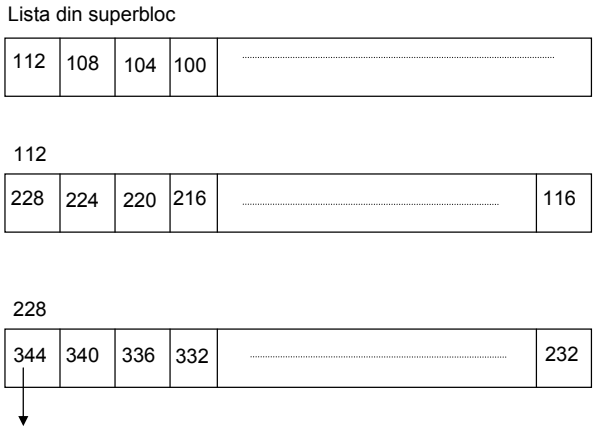
La eliberare blocurilor de date

- daca lista din superbloc nu e plina nr blocului eliberat se plaseaza in aceasta lista

- daca lista e plina blocul eliberat devine un bloc de legatura: nucleul scrie lista din superbloc in blocul eliberat si apoi scrie acest bloc pe disc.

- in superbloc se scrie numai nr blocului, acesta apărând ca singurul membru al listei

Lista de numere de blocuri libere



## Comenzi UNIX

Reprezintă ***interfața*** dintre **utilizator** şi **sistemul de operare**.

Programe care se lansează in execuție cu ajutorul unui program cu rol special ***interpretorul de comenzi - Shell.***

***Interpretorul de c-zi tradițional în UNIX a fost realizat de BOURNE (sh).***

***Interpretoare:***

***- csh ( C-shell-Berkeley);***

***- ksh ( korn Shell);***

***- bash;***

***- tcsh;***

***- zsh.***

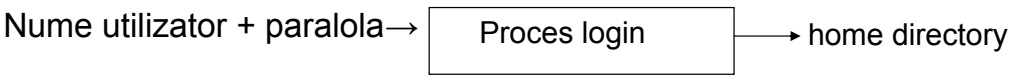
***Deosebiri:***

***- facilitățile de programare;***

***- elemente de interactivitate;***

***- modul de prez. a rezultatelor unor c-zi catre utilizator.***

Cum lucrează shell-ul




-home directory indicat în /etc/passwd

- se lansează în execuție programul al cărui nume figurează în ultimul câmp al aceleiasi intrări
- Procesul login e înlocuit cu noul program

Modul de lucru al interpretorului de c-zi

- afişează un prompt
- preia c-da şi argumentele ei şi:
  - c-da internă: o execută
  - altfel, numele c-zii e folosit pentru identificarea unui fişier executabil care e încărcat şi executat
- după terminarea c-zii reapare promptul care invită utilizatorul să introducă o nouă c-dă.

Cum lucrează shell-ul

- În Unix Shell -ul e considerat un program obişnuit, neavând prioritate mai mare ca alte procese
- Command.com - Dos

Sintaxa generală a unei c-zi:

**comanda opțiuni parametri**

**Comanda** - desemneaza programul ( numele fişierului executabil) care executa serviciul

**Opțiuni** - particularizează modul de execuție a comenzii

**Parametrii** - obiectele asupra cărora se execută c-da.

Erori:

**$ rm**

Produce:

**syntax: rm -[rtf] file(s)**

Clasificarea comenzilor:

1. **operatii asupra fisierelor si cataloagelor**
2. operatii asupra proceselor;
3. operatii de informare si/sau administrare;
4. oparatii de prelucrare a fisierelor text;
5. opeartii asupra perifericelor
6. diverse.

Manual de referinta+ Manual on line ( **man**)

**Comenzi pentru operatii generale asupra fisierelor si cataloagelor**

1. **Afisarea numelui catalogului curent** ( **print working directory**)

   **$ pwd**

   /home/geta

   **$**

2. **Afisarea continutului unui catalog** ("**list**")- numele fisierelor de diferite tipuri

   **ls [optiuni] [nume fisiere si cataloage]**

   Rezultat:

   **$ ls**

   bin

   etc

   main.c

   src

   **$**

Obs. Implicit comanda **ls** se refera la **catalogul curent** daca in sintaxa nu e prezent un nume de catalog.

Optiuni: -l (“long”)- informatii suplimentare culese din **nodul index** despre fiecare fisier descris in **catalogul** la care se refera c-da.

Exemplu:

**$ ls –l**

total 44

drwxr-xr-x 2 ij staff 512 Oct 13 11:20 bin

**OBS** – tipul fisierului: **d=** directory; **-** = fisier obisnuit *;

– bitii de protectie/acces( r= read, w = write, x = execute), listati in ordinea: proprietar, grupul propr., restul utilizatori;

– nr de leg pentru intrarea curenta, nr de intrari care indica spre acelasi fisier fizic;

– numele de login al proprietarului;

- grupul proprietarului;

– lung. fisierului in octeti;

– data ultimei modificari a fiserului;

– numele fisierului;

*

- **b:** fisier special, (un periferic), orientat pe bloc;
- **c:** fisier special orientat pe caracter;
- **l:** intrarea e o legatura simbolica;
- **p:** intrarea e un fisier special FIFO sau PIPE;
- **s:** intrarea e un soclu (“socket”) pt comunicarea in retea;

Fisiere ascunse- incep cu caracterul (**.**). Nu se listeaza cu c-da **ls**.

**Optiunea –a** (“**all**”)

**$ ls –al**

total 44

drwxr-xr-x 2 ij staff 512 Oct 13 1994 .

-rwxr-xr-x 2 root staff 1512 Aug 13 1994 bin

**Optiunea – i (“index”)** da numarul nodului index asociat fiecarei intrari din catalogul listat. Pentru exemplificare se considera subcatalogul **bin** al catalogului curent

**$ ls –li bin**

**32473 -rwxr-xr-x 1 ij staff 512 Oct 13 1994 netscape**

**3. Schimbarea catalogului curent**

Comanda**: cd (change directory)**

Sintaxa**:**

**cd [nume catalog]**

**Efect: catalogul dat ca parametru devine noul catalog.**

Utilizarea comenzii fara parametru determina din orice pozitie reintoarcerea in catalogul gazda al utilizatorului.

**4. Stergerea unei/unor intrari de catalog (stergere de fisiere)**

Comanda: **rm** (“**remove**”)

Sintaxa:

**rm [optiuni] nume_de_intrari**

**Efect**:- **se sterg doar intrari de catalog;**

- **nr de legaturi catre un fisier ( dat de nodul index al fisierului este redus cu 1);**
- **utilizatorul trebuie sa aibe drept de scriere in catalogul in care se gaseste intrarea.**

**Optiunea –i:** utilizatorul este **intrebat** inainte de stergere.

**Optiunea –r( sau R): determina si parcurgerea recursiva a subcataloagelor din catalogul la care** se refera comanda. Are semnificatie in cazul **specificarii generice de nume de fisiere.**

Obs

- nu este permisa stergerea intrarii **”..”** echivalentă cu ştergerea catalogului părinte
- daca se doreste specificarea **mai multor intrari ca argumente** la c-da **rm <=> specificarea generica** a numelor de fisiere cu ajutorul ***metacaracterelor***

| metacaracter | descriere |
|---|---|
| * | Corespunde oricarui sir de caractere (inclusiv sirul vid) |
| ? | Coresp cu orice caracter alfanumeric |
| [abc…] | Coresp cu oricare din caracterele specificate intre paranteze luate cite unul |
| \m | Coresp cu metacarcterul m, interpretat ca si caracter obisnuit |

Exemple:

- specificarea * corepunde tuturor intrarilor dintr-un catalog al caror nume **nu incepe cu “.”** OBS **“.*”** Specificarea generica a tuturor intrarilor dintr-un catalog care incepe cu **“.”;**
- specificarea ***.c** corespunde tuturor intrarilor dintr-un catalog care au sufixul **.c ( de expl. main.c, f3.c, print.c);**
- specificarea **[A-Z]***corespunde tuturor intrarilor dintr-un catalog al caror nume incepe cu o litera mare;
- specificarea **cap?** corespunde tuturor intrarilor cu nume format din 4 caractere din catalogul dat, unde primele 3 caractere sunt **”cap”** iar al patrulea este un caracter alfanumeric oarecare (cap1, cap2, cap3);

**Comanda:**

**$ rm * .txt**

**Efect:** interpretorul de c-zi **construieste linia numelor tuturor intrarilor** catalogului curent care au treminatia **.txt** si lanseaza in executie comanda **rm** transmitindu-i ca argument lista construita.

Obs: si in c-zile anterioare se poate lucra cu **nume generice**

% **ls –rl /usr/src/*.c**

Va produce lista tuturor fisierelor (intrarilor) cu terminatia **.c** (fisiere cu cod C ), din catalogul **/usr/src**.

**5. Stergerea unui catalog**

**Obs "rm"** nu afecteaza catalogul ci doar stergerea recursiva a tuturor intrarilor in catalog.

Comanda: **rmdir (remove directory)**

Sintaxa:

**rmdir [optiuni] nume_catalog**

Obs. Este permis sa fie sterse numai cataloagele care sunt goale( nu contin alte intrarui decit **.** si **..**)

## 6. Crearea unui catalog

Comanda:

**mkdir [-p] nume_catalog**

Obs. Daca este prezenta optiunea –p se creaza si cataloagele parinte care lipsesc din numele de cale dat pt catalogul curent

**$ mkdir –p /home/ij/src/c++/so/exemple**

Obs. Daca cataloagele **c++** si **so** lipsesc( nu au fost inca create) ele vor fi crate in aceasta ordine inainte de crearea catalogului **exemple**

## Comenzi pentru operatii cu fisiere

1. **Listarea continutului unui fisier**

   Comanda: cat (“catenate”)

   Sintaxa:

   **cat [optiuni] [nume_fisier]**

Efect: se aplica in special fisierelor text- pe ecran(in cazul fisierelor meri) se vede doar ultima parte;

**c-da more** – afiseaza cate un ecran la un moment dat

Optiuni:

- **n**: precede fiecare linie de text afisata cu nr ei de ordine;
- **s**: substituie in afisare mai multe linii goale consecutive cu una singura;
- **v**: afiseaza si caractere netiparibile( exceptie TAB si NEWLINE);

Obs. Daca e omis numele fisierului, implicit e **fisierul standard de intrare**: **tastatura**. Pt. terminarea executiei : EOF (CTRL-D)

**2. Copierea unui fisier**

Comanda: cp (“copy”)

Sintaxa:

cp [optiuni] nume1 nume2

Efect:

- daca ambii parametrii sunt nume de fisier obisnuite, **nume2** va fi un fisier nou( I se aloca **nod index** si **blocuri de date proprii**), cu continut identic cu cel al fisierului **nume 1**;

- daca nume 1 este un fisier obisnuit iar nume2 este un catalog, atunci in catalogul nume2 se creaza o **intrare** cu numele nume1 pt copia fisierului.

-daca atit nume1 cit si nume2 reprezinta cataloage, atunci comanda ***cp*** se poate executa doar daca este prevazuta si optiunea ***–R sau –r,*** iar efectul este crearea in nume2 a unui catalog nume1 si copierea recursiva a fisierelor din catalogul original nume1 in catalogul nou creat.

Optiuni:

-i determina cerere de confirmare daca prin copiere s-ar suprascrie un fisier existent

**3. Schimbarea numelui (pozitiei) unui fisier**

Comanda: **mv (“move”)**

Sintaxa:

**mv [optiuni] sursa destinatie**

Efect: - nu creaza fisier nou ci doar schimba numele sau sau pozitia sa intr-un sistem de fisiere;

(Obs; o operatie mai mult asupra cataloagelor)

- daca sursa si destinatie sunt nume de fisiere obisnuite atunci intrarea de catalog cu numele sursa dispare si se creaza o intrare cu numele destinatie;

- daca sursa este un nume de fisier obisnuit sau de catalog iar destinatie este nume de catalog, dispare intrarea sursa si se creaza o intrare cu numele sursa in catalogul destinatie; daca sursa este un catalog intreaga sub-ierarhie sursa isi schimba pozitia

- daca intrarea destinatie exista ea e suprascrisa. Se poate evita supra scrierea prin optiunea –i ca si la cda cp.

Obs. Nu se poate realiza traversarea sistemelor de fisiere in situatia in care sursa e un catalog ( cataloagele nu se pot muta dintr-un sistem de fisiere in altul).

**4. Crearea unei noi legaturi pentru un fisier**

Comanda: ln ("link")

Sintaxa:

ln [-fs] nume_cale [legatura] 1.

ln [-fs] nume_cale…catalog 2.

Efect: - creaza noi intrari de catalog care indica spre un anumit fisier

-1. noua intrare de catalog apare sub numele legatura;

- 1. al-II-lea arg lipseste → leg creata in **catalogul curent** are acelasi nume ca si ultima componenta din nume_cale;

- 2. in catalog se creaza legaturi avind ca **nume** ultimele componente din lista de nume_cale

Obs: inrudita cu **cp** insa nu produce si copiere fizica a fisierului
face sa creasca nr de legaturi din nodul index al fisierului

-tipuri de legaturi create:

-fixe ("hard")- implicit;

- simbolice- in prezenta optiunii –s; Sunt fisiere text al caror continut este nume_cale si al caror nume este legatura. Legaturile simbolice pot traversa sistemele de fisiere;

- nu se pot crea leg fixe spre un catalog. Doar superutilizaorul poate forta aceasta operatie prin optiunea –f ("force);

$ cd /home /ij

$ ln –s /usr/local/bin/change/ mychange

$ ls –l mychange

lrwxrwxrwx i ij staff 21 Oct 17 12:02 mychange → /usr/local/bin/change/

Obs In catalogul gazda al utilizatorului s-a creat o leg simbolica, mychange spre c-da /usr/local/bin/change/ . Lung fisierului /usr/local/bin/change/ are 21 caractere ca si nr de caract din numele de cale dat.

## Comenzi pentru operatii asupra proceselor

**1. Listarea proceselor active in sistem**

**Comanda: ps (“procesus status”)**

**Sintaxa:**

**ps [optiuni]**

**Efect: info despre procesele existente in sistem**

**$ps**

| PID | TT | STAT | TIME | COMMAND |
|---|---|---|---|---|
| 5203 | p3 | S | 0:00 | -tcsh(tcsh) %shell de login |
| 5206 | p3 | R | 0:00 | ps |
| 3849 | p9 | IW | 0:02 | -tcsh(tcsh) %shell de login |

**PID- identificatorul procesului care identifica univoc procesul pe durata existentei acestuia in sistem;**

**TT(TTY) – terminalul de control (p3, p9 – ferestre sau terminale virtuale dintr-o interfata grafica)**

**STAT: starea procesului:**

**R-running S-sleeping< I-idle> W-SWAPPED out**
**Z- zombie N-nice**

Zombie: procese terminate care asteapta ca parintele lor sa execute un apel **wait( ).**

**TIME** – timpul de procesor folosit de proces ( minute:secunde)

**COMMAND** – numele programului executat de fiecare proces

Exemplu: **-** utilizatorul are deschise 2 sesiuni (“**-**” este un shell de **login**)

OPTIUNI:

**-a** : determina afisarea de info si despre procesele altor utilizatori momentan conectati la sistem;

**-l** : determina o afisare mai extinsa ce include

**UID**

**F**: fanioane care indica tipul de operatii executate de proces;

**PPID**: identificatorul procesului parinte;

**CP**: factorul de utilizare a procesului pe termen scurt( folosit la planificarea procesului in executie);

**PRI**: prioritatea procesului in sistem;

**NI**: incrementul de planificare al procesului;

**SZ**: dimens combinata (in KB) a segmentelor de date si stiva ale procesului;

**RSS**: dimens reala in memorie a procesului (in KB);

**WCHAN**: evenim(o adresa in sist sau un nume simbolic) pe care-l asteapta procesul;

**-u:** determina afisarea rezultatelor dupa urmatorul format:

**USER PID %CPU %MEM SZ RSS TT STAT START TIME COMMAND**

Cimpuri noi:

**USER**: numele proprietarului fisierului;

**%CPU**: utilizarea procesorului de catre proces;

**%MEM**: procentul de memorie reala folosit de proces;

**START**: timpul cind procesul a fost creat( daca e in aceeasi zi) sau data crearii.

**$ ps –u**

| USER | PID | %CPU | %MEM | SZ | RSS | TT | STAT | START | TIME | COMMAND |
|---|---|---|---|---|---|---|---|---|---|---|
| Ada | 5206 | 0.0 | 2.1 | 204 | 460 | p3 | R0 | 13:26 | 0:00 | ps-u |

**-e:** determina ca la fiecare comanda sa se afiseze atit argumentele cit si ambianta de executie ( variabilele de mediu si valorile acestora);

**-x:** in informatiile afisate se includ si procesele **fara terminal de control**( procesele sistem, cum ar fi: swapper, pagedaemon, etc.)

## 2. Terminarea fortata a unui proces

**Comanda: kill**

**Sintaxa:**

**kill [optiuni] [pid]**

Argumentul **pid** e obligatoriu ( mai putin cazul cind se folosete op **–l)**

In absenta unei optiuni, c-da kill termina proceselor al caror pid e mentionat

**$ kill 5418 6230**

**OBS:** **-** procesele **trebuie sa apartina utilizatorului** care emite c-da;

- numai **superutilizatorul** poate termina orice proces din sistem;
- c-da e precedata de regula de ps pt a determina **pid-urile** proceselor care trebuiesc terminate;
- in realitate c-da trimite un **semnal** la procesele identificate. In absenta optiunilor semnalul este **TERM(“terminate”)**;
- c-da permite si transmiterea **altor semnale** a caror identitate se specifica in cimpul de optiuni, fie sub forma numerica, **fie sub forma simbolica**

Pentru obtinerea simbolurilor ce pot fi utilizate:

.

**$ kill –l**

**HUP INT QUIT ILL TRAP IOT EMT FPE KILL SEGV SYS PIPE ALRM TERM URG STOP …USR**

- Simbolurile sunt listate in ordinea identificatorilor numerici ai semnalelor
- In listarile mai recente poate aparea: **1)SIGHUP 2)SIGINT 3)SIGILL 4)SIGTRAP…….**
- **In absenta oricarei optiuni** este transmis semnalul **TERM(15),** deci c-zile

**$ kill -TERM 5418 6230**
**$ kill -15 5418 6230**
**$ kill 5418 6230**

**Sunt echivalente**

Dintre semnale sunt de remarcat:

- **HUP (1)** (“hang up”)- folosit frecvent de procesele demon pentru a citi din nou fisierelor lor de configurare;
- **KILL (9)** – cu efect sigur de terminare de proces;

## 3. Listarea activitatii diversilor utilizatori

Comanda: **w (“wat”)**

Sintaxa:

## w [optiuni] [utilizator]

Efect:

-un sumar al activitatii curente din sistem, incluzind ***procesul curent*** al fiecarui utilizator in sesiune;

- fara optiuni si parametrii c-da se refera la **toti utilizatorii** si produce o **afisare in format lung**:

**$ w**

2:42 pm up 3 days, 23:55, 5 users, load average: 0.19, 0.03.00

| User | tty | login | idle | JCPU | PCPU | what |
|---|---|---|---|---|---|---|
| romeo | ttypo | Fri 8am | 3 days | 53 | 3 | -tcsh |
| ada | ttyp2 | 12:31pm | 2:11 | | | - |

**OBS**

**prima linie afisata:**

Timpul curent, de cit timp e pornit sistemul, cati utilizatori sunt conectati la sistem, incarcarile medii calculate pe ultimele 1,5 si 15 minute.

…..**JCPU** – timpul de procesor folosit de procesele curente;

**PCPU** – timpul si argumentele procesoarelor curente;

**what** - numele si argumentele procesoarelor curente;

Optiuni:

**-h**(“header”) –lipseste prima linie

**-s**(“short”) – determina omiterea coloanelor: numele terminalului, timpii de login, si procesor si argumentele comenzilor.

## 4. Listarea dinamica a proceselor din sistem

Atit **ps** cit si **w** dau o imagine statica, la un anumit moment asupra activiatii din sistem.

**C-da top afiseaza procesele din sistem si actualizeaza periodic aceasta informatie.**

Informatia afisata variaza de la o implementare la alta. Perioada de actualizare e implicit 5 secunde dar poate fi modificata printr-o optiune.

Obs- c-da **top** recunoaste citeva “**subcomenzi**” (sub forma unor caractere unice):

- **h** lista subcomenzilor acceptate;
- **q** termina executia lui **top;**
- **k** termina executia unui proces; trebuie specificat PID-ul procesului;

## Comenzi de informare si administrare

### 1. Afisarea datei si orei curente

**Comanda: date**

**Sintaxa:**

**date [yymmddhhmm[.ss]]**

In varianta fara argument c-da afiseaza data si ora curenta a sistemului:

**$ date**
**Tue Oct 15 08 36:42 EET 2007**

Obs- varianta cu argument a c-zii este accesibila numai superutilizatorului si serveste la fixarea orei si datei curente.

### 2. Afisarea utilizatorilor aflati momentan in sesiune

**Comanda: who**

**Sintaxa:**

**who**

Efect: produce o lista a utilizatorilor curenti din sistem fara detalii despre procesele acestora:

**$ who**

```
ioana   console   Oct 15    08:45
ada     ttyp0     Oct 17    12:45
mihai   ttyp1     Oct 12    12:45
```

- Afiseaza doar terminalul la care sunt conectati utilizatorii si **data inceperii sesiunii curente.**
- Info sunt extrase dintr-un fisier sistem: **/etc/utmp/** (/var/run/utmp in alte implementari.
- O varianta: whoami – pt a afla numele de login sub care lucreaza momentan utilizatorul. Este utila in combinatie cu su.

Pt sistemele care lucreaza in **retele locale**:

**rusers**- produce o lista cu toti utilizatorii momentan conectati la toate sistemele din reteaua locala.

**$ rusers**

tempusII1 mihai ada ada ada ada
tempus5 romeo ioana ada ada fane

- Accepta ca argument un **nume de calculator**.
- O simpla lista de utilizatori de la acelasi calculator de la care este emisa c-da este obtinuta prin **users.**

**$ users**

alex ada ioana mihai

### 3. Afisarea utilizatorilor recenti ai sistemului

**Comanda: last**
**Sintaxa:**

**Last [-number] [-f filename] [name]….] [tty]**

Efect ( fara optiuni si argumente): lista sesiunilor inregistrate in **fisierul sistemului /var/adm/wtmp (sau /var/log/wtmp)**

Optiuni:

- **number** permite limitarea listei la ultimele number sesiuni.
- **f** filename specifica un alt fisier decit **wtmp** in care sa se caute inregistrarile;
- Prin **name**… pot fi identificate numele de login pt care se doreste listarea, iar prin **tty**…, terminalele care prezinta interes.

**$ last -3 console**

ada console :0 Thu Oct 30 14:50 still logged in

misu console :0 Tue Oct 29 09:04 – crash (2+05:42)

root console :0 Tue Oct 30 14:50 - 09:04 (00:09)

S-a cerut listarea celor mai recente 3 intrari in sesiune de la consola sistem.

**$ last -3**

**ftp7349 ftp tempus8.cs.utt.ro Thu Oct 30 15:55 still loged in**

**ttyp6 ada tempus5.utt.ro Thu Oct 30 15:02 -15:31 (00:29)**

**……..**

Obs- c-da e rulata pe un sistem care e si server de fisiere cu acces anonim( prima linie ftp7349 reprez un astfel de transfer)

**$ last -2 reboot**

reboot system boot Thu Oct 10 14:56

reboot system boot Fri Oct 4 15:26

Numele de login reboot poate fi folosit pt a afla intrarile din **wtmp** referitoare la ultimele incercari ale sistemului dupa caderi.

**4. Comenzi de informare despre identitatea calculatoarelor din retea**

**Comanda: hostname**

Efect: numele sub care e cunoscut un calculator in retea:

**$ hostname**

B219

**Comanda: hostid**

Efect: Identificatorul unic al controlerului de retea continut in calculator("placa Ethernet"):

**$ hostid**

E2c1c808

**Comanda: domainname**

Efect: Numele **domeniului** in care se afla calculatorul:

**$ domainename**

upt.ro

## 5. Comenzi mount/umount

Perechea de c-zi reprezinta mecanismul prin care se construieste **ierarhia de fisiere** a unui sistem Unix.

Fara parametrii, c-da **mount** listeaza sistemele de fisiere **momentan montate** si **punctele de montare** ale acestora:

**$ mount**

/dev/sda3 on / type ext2 (rw)

/none on/ proc type proc (rw)

/dev/sda4 on / home type ext 2 (rw)

/dev/sda1 on / dos type msdos (rw)

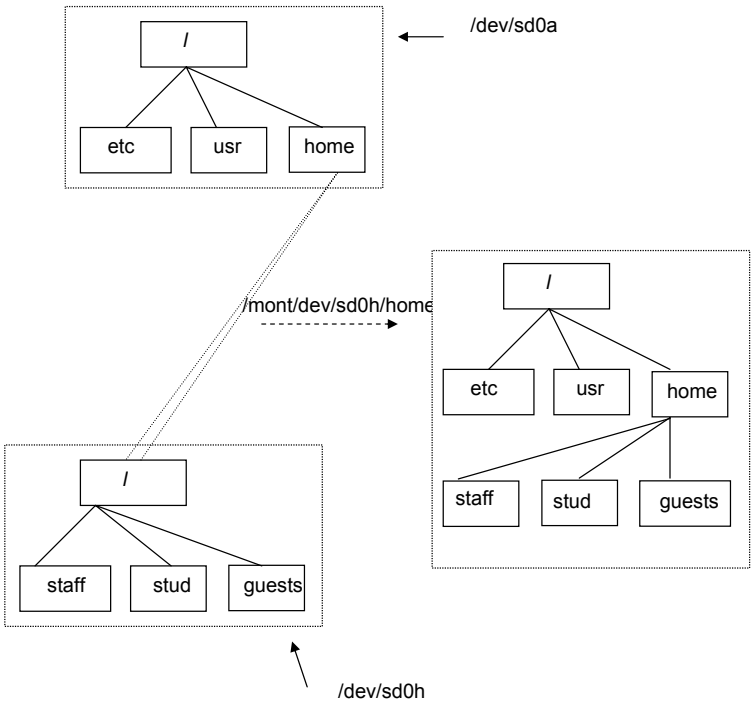
none on /dev/pts/ type devpts (rw, mode=0622)

bigfoot:/home/users/mail on/var/spool/mail type nfs (rw,addr=193.226.12.13)

Se apreciaza: **a) tipul sistemului de fisiere:**

- **ext2** (linux), **proc** (Unix SVR4- pt acces special la procese), **msdos** pt accesul din Unix la partitii (periferice) formatate DOS, **nfs** pt sisteme montate in retea**.**

**b) modul de montare: "rw"(read/write) -** utilizatorii pot crea fisiere acolo unde au drepturi;




Se prezinta 2 sisteme de fisiere create in 2 partitii ale unitatii de disc sd0

- partitia sd0a contine sistemul de fisiere radacina, in care s-au evidentiat 3 cataloage
- partitia sd0h contine catalogul radacina cu cele 3 subcataloage destinate pastrarii cataloagelor gazda a tri clase de utilizatori

In urma op de montare

**# mount /dev/sd0h/home**

catalogul radacina al partitiei /dev/sdh0 acopera catalogul /home din partitia /dev/sd0a. Daca anterior existau fiisiere in /home acestea nu mai sunt acum vizibile, iar c-da ls/home va arata ca exista cataloagele staff, stud si guests.

C-da de demontare are in mod uzual sintaxa:

**unmount sistem-fisiere | catalog**

OBS permite fie specificarea ca parametru a unui sistem de fisiere, de fapt nume de fisier special, fie a unui punct de montare.

Demontarea sistemului de fisiere montat in exemplul anterior:

# unmount /dev/sd0h

Sau:

# unmount /home

Nu este permisa demontarea unui sistem de fisiere atita timp cit catalogul curent este in cadrul acelui sistem.

## 6. Afisarea spatiului liber ocupat intr-un catalog sau sistem de fisiere

**Comanda: du (“disk usage”)**

**Sintaxa:**

**du [-s] [-a] [nume fisier….]**

Efect:- versiunea BSD a c-zii afiseaza spatiul ocupat in kocteti;

- versiunea System V afiseaza nr de blocuri de 512 octeti;

-daca un fisier are legaturi multiple este luat in considerare o singura data

Optiuni:

**-s** afisarea doar a totalului general pt numele de fisiere date ca argumente;

**-a** genereaza cite o intrare pt fiecare fisier

- daca nu apare nici o optiune se afiseaza totalul numai pentru cataloage

- in absenta argumentelor (nume de fisier) c-da se aplica asupra catalogului curent;

Fie catalogul cu continutul:

**$ ls –ali**

```
Total 163
134555 drwxr-xr-x   3  ada     512  Oct 25  14:20  .
216317 drwxr-xr-x  37  ada    2560  Oct 25  14:18  ..
134556 –rw-r--r--   2  ada   80208  Oct 25  14:19  art.ps.gz
134557 drwxr-xr-x   2  ada     512  Oct 25  14:20  subdir
134556 –rw-r--r--   2  ada   80208  Oct 25  14:19  tr92.ps.gz
```

Obs: exista 2 legaturi spre acelasi fisier in ac catalog: nodul index 134556 apare in ambele legaturi, iar **ls** raporteaza la total suma lungimilor afisate pentru toate intrarile.

C-da

**$ du –a**

```
79    ./tr92.ps.gz
79    ./subdir/art.ps.gz
80    ./subdir
160   .
```

Examinarea acestui rezultat arata ca spatiul ocupat de fisierul cu 2 legaturi este raportat o singura data (sub numele legaturii care a fost creata prima)

Afisarea sp liber si ocupat pt un intreg disc (sau partitie) se realizeaza cu c-da **df**

**df [-a] [-i] [-t type] [filesystem….] [filename…]**

Efect: in absenta optiuni si argumente c-da afiseaza situatia spatiului pentru toate sistemele de fisiere montate:

$ df

| Filesystem | kbytes | used | avail | capacity | Mounted on |
|---|---|---|---|---|---|
| /dev/sd0a | 7735 | 5910 | 1052 | 85% | / |
| /dev/sd0g | 151339 | 128358 | 7902 | 94% | /usr |
| Tempus0:/home/tempus | 492613 | 410853 | 32499 | 93% | /home/tempus |
| Tempus0:/usr/local | 492613 | 408244 | 35108 | 92% | /usr/local |

Obs:

"used" + "avail" < "kbytes"

daca se folosesc argumente **filesystem** sau **filename** raportul se refera numai la sistemele de fisiere prezente in comanda.

implementarea pt System V a c-zii **df** afiseaza doar spatiul disponibil (in unitati de 512 octeti) si nr de i-noduri libere.

**7. Controlul drepturilor de acces la un fisier**

**Comanda: chmod ("change mode")**

**Sintaxa:**

**chmod [-fR] mode filename**

Efect:

- specificarea modului se face numeric sau simbolic;

- **numeric:** se redau in octal bitii corespunzatori pt fiecare din cele 3 categorii de utilizatori. Astfel 766 este echivalent cu rwxr-xr-x. Exista si o a patra cifra octala ( se scie prima) care corespunde, in ordine, bitilor :"setuid", "setgid", si "sticky".

In cazul specificarii simbolice, modul dorit se reda in forma:

**[ cine] op drept [op drept….]**

unde cine e o combinatie de:

u – "user"

g – grup

o – "others" (ceilalti utilizatori)

a – "all" (toti), echivalent cu combinatia ugo.

Daca se omite cine, valoarea implicita este **a**.

Operatiile **op** trebuie sa fie:

+ pt a adauga un drept

- pt a sterge un drept

= pt a fixa explicit un drept

Drepturile se specifica prin literele:

- r-read
- w-write,
- x- execute
- X – da drept de executie daca fisierul specificat este catalog sau daca alte clase de utilizatori au drept de executie.
- s – specifica “setuid” (impreuna cu **u** ) sau “setgid” (impreuna cu **g** );
- t- specifica “sticky bit”;

Optiuni ale c-zii:

- f “force” – nu vor apare mesaje de eroare daca nu se poate schimba modul unui fisier;
- R- “recursiv”, c-da se aplica recursiv pt toate fisierele cu numele indicate

**$ ls –l**

**total 479**

**-rwxr-xr-x 1 ada 401408 oct 26 11;53 tar**

**-rw-r--r-- 1 ada 80208 oct 25 11;53 tr92.ps.gz**

Pt ca fisierul executabil tar sa devina “setuid”, adica pe durata executiei utilizatorul sa fie proprietarul lui se da c-da

**$ chmod u+s tar**

**Rezultatul:**

$ ls –l tar

-rw**s**r-xr-x 1 ada 401408 oct 26 11;53 tar

**8. Modificarea proprietarului unui fisier**

**OBS** Doar superutilizatorul poate schimba dreptul de proprietate prin comanda:

**$ chown [fHR] owner[.group] filename**

Atit owner cit si grup se pot specifica fie numeric (uid, gid) fie simbolic.