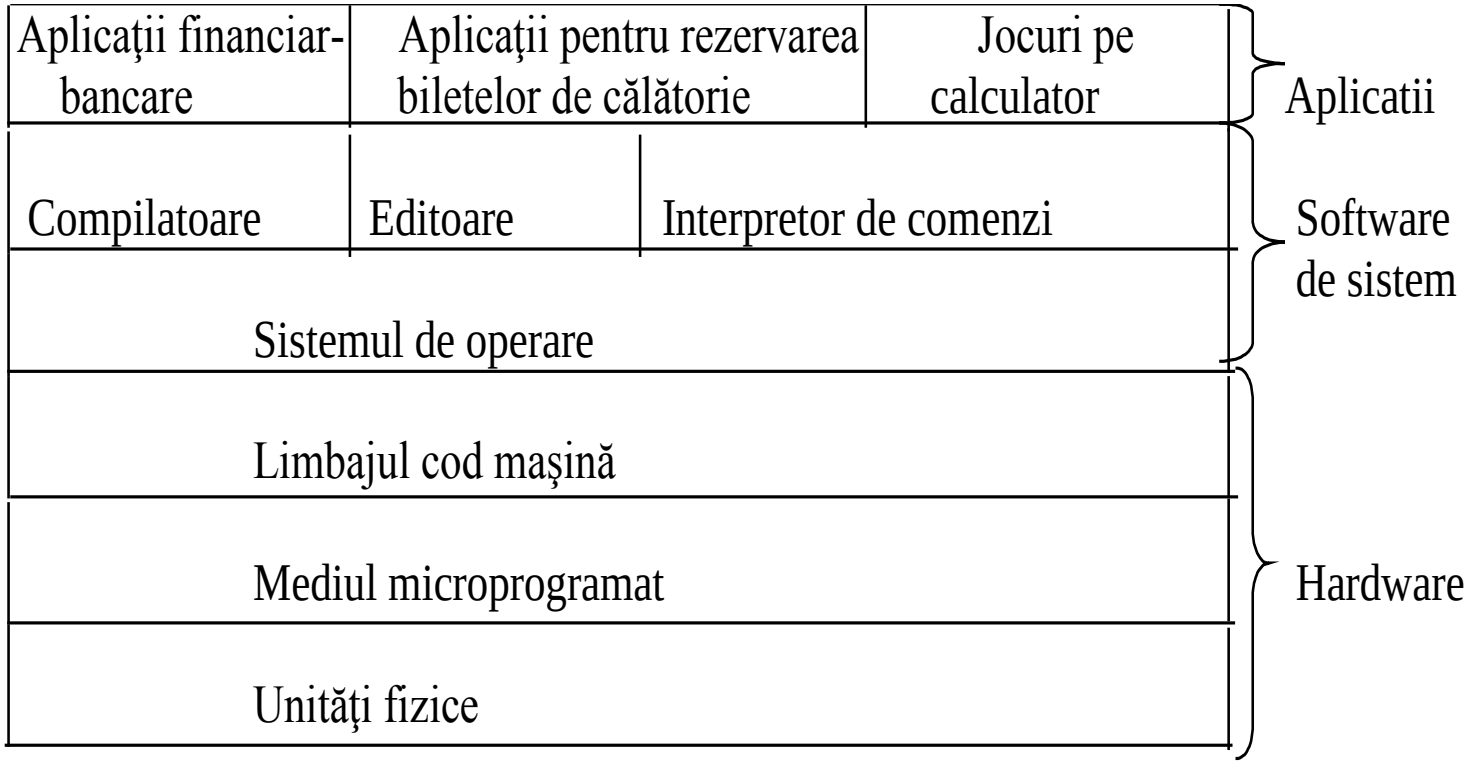
INTRODUCERE

Obiectivele cursului:

- evidenţierea rolului central al sistemului de operare în cadrul componentei software a unui sistem de calcul
- prezentarea evoluţiei sistemelor calcul şi a celor de operare;
- prezentarea conceptelor care stau la baza sistemelor de operare, care vor fi dezvoltate în capitolele următoare;
- prezentarea structurii sistemelor de operare şi a funcţiile lor;
- prezentarea componentelor hardware care realizează interfaţa cu sistemul de operare.

Locul sistemului de operare în cadrul unui sistem de calcul

- Componentele principale ale unui sistem de calcul (SC) sunt cea fizică(hardware) şi cea logică(software). La rândul ei, componenta logică este formată din software-ul (programele) de aplicaţii şi software-ul de sistem, aşa cum este redat în figura următoare.



- Programele de aplicaţii au scopul de a rezolva cu ajutorul calculatorului o problemă specifică dintr-un anumit domeniu de activitate (financiar-bancar, ştiinţific etc.). Programele de sistem oferă contextul (mediul) în care programatorii pot crea propriile programe de aplicaţii, care nu sunt disponibile la nivel fizic. Sistemul de operare (SO) face parte din componenta soft de sistem.
- Este dificil să se dea o definiţie completă a ceea ce este un SO, în schimb, este mult mai uşor să se vadă ce face un SO. Astfel:

  - SO oferă facilităţile necesare unui programator pentru a-şi construi propriile aplicaţii.

  - SO gestionează resursele fizice(memorie, discuri, imprimante etc) şi cele logice (programe de sistem, fişiere, baze de date etc) ale sistemului de calcul, oferind posibilitatea ca utilizatorii să poată folosi în comun aceste resurse, pe baza unor anumite reguli, ceea ce conduce la scăderea cheltuielilor de prelucrare şi la creşterea performanţelor sistemului de calcul.

  - SO oferă o interfaţă prin care aplicaţiile utilizator şi cele de sistem au acces la componenta hardware.

Evoluţia sistemelor de operare

- **generaţia I:** dispun numai de echipamentul hard.
- **generaţia a II-a** :

- componente hardware noi (cititorul de cartele, imprimanta, banda magnetică)
- componente software (compilator pentru limbajul FORTRAN, programe specializate destinate asamblării, încărcării şi înlănţuirii programelor, biblioteci software de funcţii uzuale, rutine de interfaţă cu componentele fizice numite **drivere** etc.).
- conceptul de **monitor**, ce reprezintă o formă rudimentară de SO. Lucrările (job-urile) erau o succesiune a fazelor (**editarea** textului sursă al programului, **compilarea, editarea legăturilor**, **depanarea, execuţia** programului).

- **Monitorul rezident.** Intervenţia umană între faze presupune o mare pierdere de timp şi o utilizare ineficientă a echipamentului. Pentru a evita aceste neajunsuri, a apărut conceptul de **încărcare automată a job-urilor si a fazelor**.
- Pentru aceasta a fost creat un mic program, denumit **monitor rezident**, care realiza acest lucru. Programatorul trebuie să insereze, printre cartelele programului şi ale datelor sale, unele cartele speciale numite **cartele de comandă** care se adresează acestui monitor rezident. Prin această regulă de diferenţiere, s-a definit de fapt un **limbaj de control** al job-urilor**.** Prin intermediul lui, se comandă trecerea de la o fază la alta sau de la un job la altul.
- Monitorul rezident este programul permanent activ. Pentru desfăşurarea înlănţuirilor de faze, este necesar ca operaţiile de I/O să fie făcute de către monitor şi nu de către programul utilizator, pentru a depista cartelele care îi sunt adresate, adică cele de comandă.
- Tipul de SO prezentat se numeşte **sistem serial,** deoarece job-urile se execută unul după altul(**prelucrare pe loturi** - **Batch processing**).
- O altă caracteristică a acestui sistem este **monoprogramarea**, adică CPU nu se ocupă de alt job până când nu-l termină pe cel curent. De asemenea, programului utilizator nu-i este permis să modifice zona de memorie a monitorului rezident şi nici să comande oprirea întregului sistem.

- **Conversiile off-line** constau în transferarea conţinutul cartelelor perforate pe benzi magnetice. Pentru aceasta se foloseau nişte dispozitive de conversie relativ simple. Conţinutul benzilor era utilizat de către SC în locul cartelelor perforate, unitatea de bandă fiind intrarea standard. Analog, rezultatele, sub forma unor linii de imprimantă, sunt depuse mai întâi într-un fisier pe un suport magnetic, iar la terminarea lucrului, conţinutul fişierului este listat la imprimantă. De asemenea, prin această metodă, se reduce timpul de lenevire al CPU datorat vitezei de lucru mai mici a unor periferice.
- Odată cu operarea off-line a apărut şi noţiunea de **independenţă faţă de dispozitiv** a programelor, adică aceeaşi operaţie de I/O să poată fi realizată de pe diferite dispozitive fizice. Acest lucru se realizează prin aşa zisele **dispozitive logice de I/O**, ce reprezintă nişte identificatori utilizaţi de programe, care sunt asociaţi prin intermediul SO dispozitivelor fizice.
- Utilizarea zonelor tampon (buffere) este un alt concept utilizat de SO în scopul utilizării eficiente a CPU. Informaţiile de pe discuri care vor fi prelucrate de CPU sunt aduse în zone de memorie internă numite buffere.

- **Generaţia a III-a de calculatoare.**
- Conceptul de **multiprogramare** reprezintă modul de exploatare a unui SC cu un singur procesor central, care presupune existenţa simultană în memoria internă a mai multor programe, care se execută concurent.
- CPU este componenta hardware a calculatorului formată îndeosebi din componente electronce, pe când unităţile de I/O sunt formate din componente mecanice. Deci, pe de o parte CPU este mai scumpă, iar pe de altă parte ea lucrează mult mai rapid decât unităţile de I/O. Astfel, s-a pus problema utilizării cât mai eficientă a unităţii centrale a calculatorului.
- Multiprogramarea a rezolvat această probleme. Pe scurt, lucrul în multiprogramare se desfăşoară astfel:

  - în fiecare moment CPU execută o instrucţiune a unui program(starea RUN);

  - restul programelor, fie că aşteaptă apariţia unui eveniment extern, de exemplu terminarea unui I/O etc. (starea WAIT), fie că sunt pregătite pentru a fi servite în orice moment de către CPU (starea READY).

  - Trecerea unui program din starea RUN în starea WAIT este realizată de către program, în momentul când trebuie să execute o instrucţiune de I/O.

  - Trecerea programelor din starea RUN în starea READY şi invers este realizată de către SO pe baza unui algoritm de planificare, concept pe care îl vom detalia într-o secţiune următoare.

- **Canalul de intrare – iesire.**
- Pentru a creşte gradul de exploatare al CPU prin eliminarea timpilor de aşteptare al acesteia, datoraţi diferenţei între vitezele de execuţie a echipamentele electronice ale CPU şi celor mecanice ale dispozitivelor de I/O, a apărut, la nivelul tehnologiei hard existente la vremea respectivă **canalul de intrare – iesire.**
- Acesta este un procesor specializat pe operaţii de I/O, care poate funcţiona în paralel cu CPU. Pentru a fi lansat în execuţie, canalul primeşte de la CPU o comandă de efectuare a unei operaţii de I/O. După lansare, cele două procesoare îşi continuă activitatea în paralel.
- Sincronizarea între CPU şi canalul de intrare – ieşire se poate realiza prin t**estarea periodica (pooling) a perifericelor de către CPU** sau **printr-o întrerupere lansată de către periferic**.
- O î**ntrerupere** este o rutină aflată la o adresă fixă de memorie lansată în urma apariţiei unui semnal hard, care perturbă execuţia firească a instrucţiunilor programului în curs. Modul de lucru al întreruperilor va fi prezentat mai târziu. Printre altele, prin intermediul lor se realizează comunicarea dintre SC şi dispozitivele lui periferice (discuri, imprimante etc.). Rutina de întrerupere determină perifericul care a emis întreruperea şi, eventual dă perifericului o nouă comandă de I/O.8

- Relaţia dintre CPU şi un dispozitiv periferic se realizează astfel:
- fiecare periferic conţine o zonă tampon proprie, capabilă să păstreze o înregistrare (o linie de imprimantă, imaginea unei cartele).
- CPU, printr-o rutină de I/O, acţionează dual, în funcţie de operaţia efectuată. În cazul scrierii, pune din memorie informaţii în această zonă tampon, iar în cazul citirii preia informaţiile şi le depune în memorie.
- Dispozitivul periferic acţionează şi el dual, în funcţie de operaţia care îi este comandată.
- În cazul scrierii, ia informaţiile din zona tampon proprie şi le depune pe suport. În cazul citirii, ia informaţiile de pe suport şi le depune în zona tampon proprie.
- În prezent, SC folosesc două tipuri de canale: **selector** si **multiplexor**. Canalul selector este destinat să realizeze schimbul dintre memorie şi perifericele rapide (discuri), care lucrează, la un moment dat cu un singur periferic. Canalul multiplexor este capabil să lucreze simultan cu mai multe dispozitive periferice.

- **SPOOLING - Simultaneous Peripherical Operation On-Line.**
- Au apărut discurile magnetice, care au permis operarea on-line, simultan, cu mai multe periferice (SPOOLING). El s-a obţinut prin îmbinarea utilizării zonelor tampon multiple cu conversiile off-line şi cu multiprogramarea.
- SPOOLING funcţionează astfel:

  - Se citesc de la un cititor, cartelele care compun un job.

  - Când jobul a fost citit complet, imaginile cartelelor sunt depuse într-un buffer pe disc. Un astfel de job spunem că se află în starea HOLD.

  - Când CPU este liber, el alege unul dintre joburile aflate în stare HOLD şi-l lansează în execuţie.

  - Liniile “tipărite” de jobul în execuţie sunt depuse într-un buffer pe disc.

  - Atunci cand jobul a fost executat complet, bufferul lui de ieşire pe disc devine disponibil pentru listare. Spunem că jobul se află în starea FINISH.

  - Conversia de ieşire se lansează automat, listând pe o imprimanta bufferul unui job aflat în starea FINISH.

- Operaţiile de conversie se fac în paralel cu execuţia în regim de multiprogramare. Dacă pentru execuţie sunt disponibile mai multe zone ale memoriei interne(partiţii), atunci toate au acces la cozile HOLD si FINISH, executandu-se în paralel mai multe joburi. De remarcat la tehnica SPOOLING este **simultaneitatea**. La un moment dat sistemul are în lucru trei categorii de joburi:

- joburi în **curs de citire** de la unul sau mai multe cititoare; aceste joburi sunt trecute în coada HOLD.
- joburi în **curs de execuţie;** numărul lor poate fi cel mult egal cu numărul de partiţii disponibile sp lucreze în multiprogramare;
- joburi în curs de listare; numărul lor poate fi cel mult egal cu numărul de imprimante active conectate la sistem; aceste joburi sunt preluate din coada FINISH.

- Această tehnică se foloseşte şi la calculatoarele actuale pentru listările la imprimantă.
- Dintre sistemele de operare specifice acestei generaţii de calculatoare, cele mai reprezentative sunt sistemul SIRIS pentru calculatoarele IRIS 50, care s-au produs şi în ţara noastră sub denumirea FELIX 256/512/1024 şi sistemul OS/360 pentru calculatoarele IBM 360.

- **Generaţiei a-IV-a de calculatoare**.
- **Sistemele interactive**:Un sistem interactiv permite comunicarea on-line dintre utilizator şi sistem.

  - De regulă, utilizatorul are la dispoziţie un terminal cu tastatură şi ecran, prin care comunică cu sistemul.

  - În astfel de sisteme, utilizatorul dă comanda, asteaptă răspunsul şi, în funcţie de rezultatul furnizat de comanda precedentă, decide asupra noii comenzi.

  - El poate, astfel să experimenteze uşor şi să vadă rezultatele imediat. SO pentru SC interactive conţin (cel puţin) câte un editor de texte pentru corectarea programelor sursă şi (cel puţin) câte un depanator interactiv care poate asista alte programe în execuţie.

  - Spre deosebire de sistemele seriale, sistemele interactive au un timp de răpuns rezonabil de ordinul secundelor, eventual al minutelor.
- **Time-sharing** (timp partajat) este o variantă a multiprogramării.

  - A apărut la generaţia a III-a de calculatoare (sistemul CTSS dezvoltat de MIT, sistemul MULTİCS (MULTiplexed Information and Computing Service şi mai ales sistemul UNIX .

  - A fost dezvoltat pe scară largă de generaţia a IV-a de calculatoare. Ele au fost proiectate pentru a permite mai multor utilizatori să fie conectaţi simultan la acelaşi sistem de calcul. Acest sisteme îmbină **interactivitatea** şi **multiprogramarea**.

Terminal virtual.

- Fiecare utilizator stabileşte o sesiune cu sistemul, prin intermediul unui **terminal virtual**, care este o simulare a hardware-ului calculatorului şi este implementat de către SO.

- Sistemul comută rapid de la un program la altul, înregistrând comenzile solicitate de fiecare utilizator prin terminalul său.

- Deoarece o tranzacţie a utilizatorului cu sistemul necesită un timp de lucru mic al CPU, rezultă că într-un timp scurt, fiecare utilizator este servit cel puţin o dată. În acest fel, fiecare utilizator are impresia că lucrează singur cu sistemul.

- Dacă sistemele seriale încearcă să optimizeze numărul de job-uri prelucrate pe unitatea de timp, sistemele timesharing realizează o servire echitabilă a mai multor utilizatori, aflaţi la diverse terminale.

Diferenţiere între noţiunile de job şi proces (program în execuţie).

- În sistemele timesharing, la acelaşi moment un job poate executa două sau mai multe procese, pe când în sistemele seriale un job presupune un singur proces.

- Într-un sistem timesharing multiprogramat procesele se mai numesc şi **task**-uri iar un astfel de calculator se mai numeşte şi sistem **multitasking**.

Redirectarea si **legarea in pipe.** A apărut la calculatoarele din generaţia a III-a, dar a fost dezvoltat şi utilizat îndeosebi de calculatoarele din generaţia a IV-a. Aplicarea lor presupune că fiecare program lansat de la un terminal are un **fişier standard de intrare** şi un **fişier standard de ieşire.** De cele mai multe ori acestea coincid.cu tastatura, respectiv cu terminalul de la care se fac lansările. Redirectarea intrărilor standard (intrare sau/şi ieşire) permite utilizatorului să înlocuiască intrarea standard cu orice fişier, respective să se scrie rezultatele, afişate de obicei pe ecran într-un fişier oarecare, nou creat sau să fie adăugate la un fişier deja existent. Informaţiile de redirectare sunt valabile din momentul lansării programului pentru care se cere acest lucru şi până la terminarea lui. Dupa terminarea programului se revine la fişierele standard implicite.

Dintre SO cele mai cunoscute ale generaţiei a IV-a de calculatoare, amintim sistemele DOS, Windows, Unix pentru calculatoare personale şi pentru reţele de calculatoare, sistemele RSX pentru minicalculatoare (calculatoare mainframe).

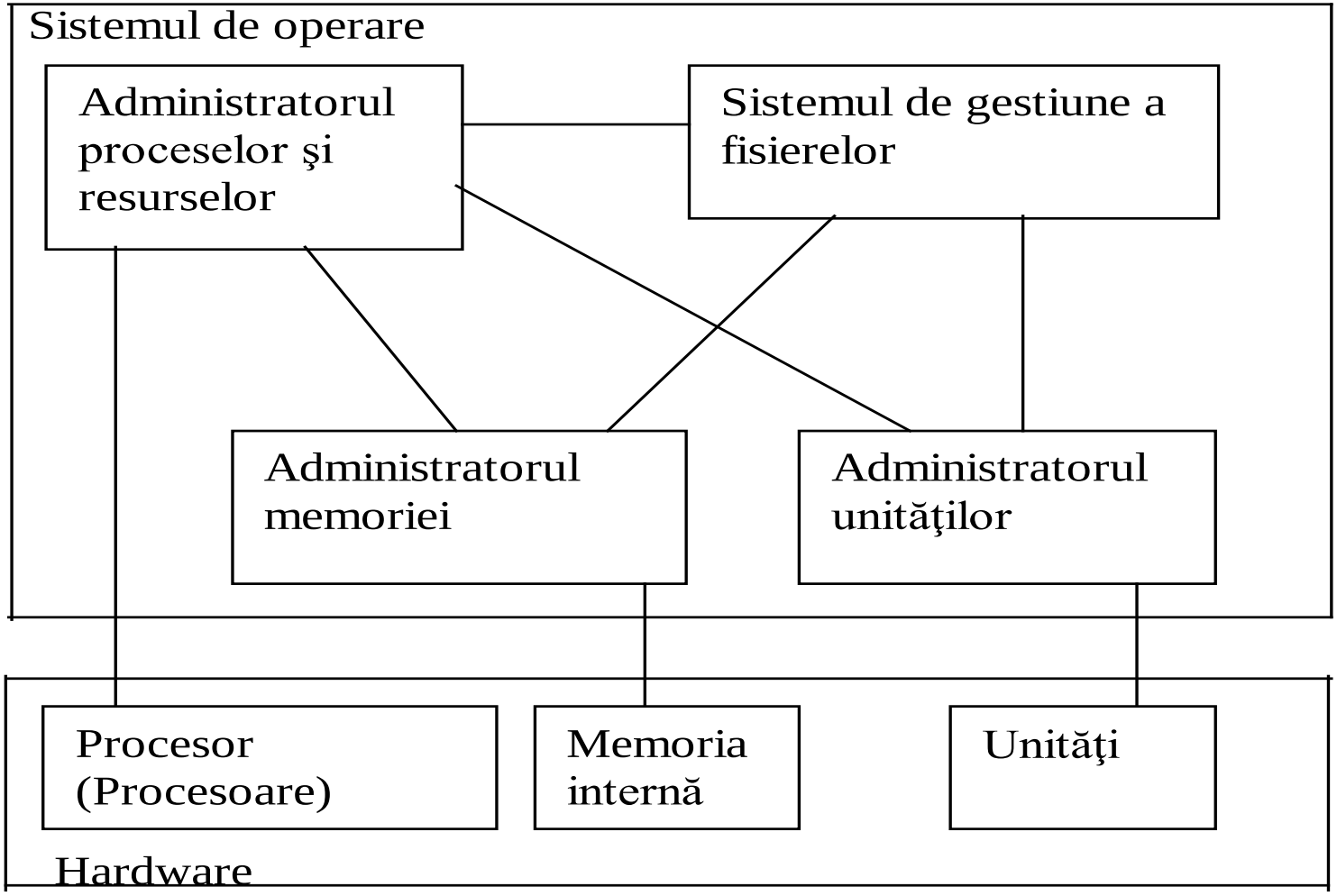
Conceptul general de proces

- Dacă sistemele de operare cu prelucrare în loturi executau lucrări(job-uri), sistemele de operare moderne bazate pe divizarea timpului execută task-uri. Aceste două concepte corespund termenului de proces. În cadrul sistemelor de operare, procesul reprezintă o entitate activă, un program în execuţie ale cărui instrucţiuni sunt parcurse secvenţial şi executate de către unitatea centrală a calculatorului. Dacă prin program înţelegem o entitate statică, o codificare într-un anumit limbaj de programare a unui algoritm, sub forma unui fişier stocat pe un suport extern de informaţie, prin proces definim o entitate dinamică, încărcat în memoria internă a sistemului de calcul. Unul sau mai multe procese pot fi asociate unui program, dar ele sunt considerate entităţi distincte.
- Orice proces foloseşte mai multe resurse ale sistemului de calcul: **contorul de program** (PC – **P**rogram **C**ounter) este un registru al UC care conţine adresa de memorie a următoarei instrucţiuni care urmează să fie executată; **stiva de program** conţine date utilizate în execuţie, parametri ai subprogramelor, adrese de retur şi alte variabile locale; **secţiunea de date**, conţine variabile globale; **timpul de lucru** al UC; **fişiere**; **dispozitive de I/O** etc. Aceste resurse sunt alocate procesului fie în momentul creerii lui, fie în timpul execuţiei lui

- În multe sisteme, procesul este unitatea de lucru. Astfel de sisteme constau dintr-o colecţie de procese. Sistemul de operare, ca şi componentă soft, este format din mai multe procese, care execută codul de sistem, pe când procesele utilizator execută programe scrise de aceştia. Toate aceste procese se execută concurent la nivelul unui sistem de calcul, sau al unei reţele de calculatoare. În mod tradiţional un proces conţinea un singur fir de execuţie; sistemele de operare moderne, privesc un proces ca fiind format din unul sau mai mute fire de execuţie.
- Execuţia proceselor este un mecanism combinat hard şi soft. Componenta software care este implicată în administrarea proceselor şi firelor de execuţie este sistemul de operare, ale cărui sarcini sunt:
- crearea şi distrugerea proceselor şi firelor de execuţie;
- planificarea proceselor;
- sincronizarea proceselor;
- comunicarea între procese;
- manipularea interblocării proceselor.

Funcţiile de bază ale sistemelor de operare

- Aceste funcţii cooperează între ele pentru a satisface cerinţele utilizatorilor. În figura urm. sunt prezentate interacţiunile între modulele care realizează funcţiile SO, precum şi între aceste module şi componentele hard ale sistemului de calcul.



Administrarea proceselor şi a resurselor

- Procesul reprezintă unitatea de bază a calculului, definită de un programator, iar resursele sunt elemente ale mediului de calcul necesare unui proces pentru a fi executat. Crearea, execuţia şi distrugerea proceselor, comunicarea între ele, împreună cu alocarea resurselor după anumite politici, sunt aspecte deosebit de importante care vor fi discutate în detaliu în cursurile următoare.

Gestiunea memoriei.

- SO alocă necesarul de memorie internă solicitat de procese şi asigură protecţia memoriei între procese. O parte este realizată prin hard, iar o parte prin soft. Această problemă va fi obiectul cursurilor următoare.

Gestiunea fişerelor

- **SO conţine o colecţie de module(SGF) prin intermediul cărora se asigură deschiderea, închiderea şi accesul utilizatorului la fişierele rezidente pe diferite suporturi de informaţii. Componentă de bază a SO, este cel mai des invocată de către utilizator şi de către operator.**

Administrarea unităţilor

- **Se referă la modul în care discurile, terminalele, imprimantele, procesoarele, memoria şi alte componente hardware sunt alocate, protejate în timpul alocării şi partajate după anumite reguli. Accesarea acestor componente se realizează prin intermediul driverelor**.

Arhitectura Von Neumann

- Sistemele de calcul se bazează pe conceptul de arhitectură Von Neumann, conform căruia partea de hardware este formată din:

►**Unitatea centrală de calcul** (CPU - Central Processing Unit), compusă din **unitatea aritmetică** şi **logică** (ALU-Arithmetical-Logical Unit) şi **unitatea de control**.

►**Unitatea de memorie primară** sau **executabilă** sau **internă**.

►**Unităţi de I/O**.

- Toate unităţile sunt conectate folosind o **magistrală** (bus), care se împarte într-o **magistrală de date** şi una **de adrese.**
- Pentru memorie şi alte unităţi se poate adăuga o magistrală de I/O, care nu este folosită de CPU. O astfel de organizare, permite ca o unitate să poată citi/scrie informaţii din memorie, fără alocarea CPU. Fiecare magistrală poate fi gândită ca fiind formată din mai multe linii (fire) paralele, care pot păstra o cifră binară.

Unitatea aritmetică şi logică

- Conţine, pe lângă unitatatea funcţională un număr de **regiştri generali** şi de **stare**. Regiştrii generali sunt folosiţi în efectuarea operaţiilor aritmetice şi logice, atât pentru memorarea operanzilor încărcaţi din memorie, cât şi a rezultatului operaţiei, care apoi va fi salvat într-o locaţie de memorie.
- CPU extrage şi execută instrucţiunile cod maşină ale procesului încărcat în memoria internă. În acest sens, CPU conţine:
- ►o componentă care extrage o instrucţiune memorată într-o locaţie de memorie;
- ►o componentă care decodifică instrucţiunea;
- ►o componentă care se ocupă de execuţia instrucţiunii, împreună cu alte componente ale SC.
- **Regiştrii** contor de program **PC (Program Counter),** respectiv registrul instrucţiune **IR (Instrucţion Register),** conţin adresa de memorie, respectiv o copie a instrucţiunii în curs de prelucrare**.**

Unităţile de I/O

- Unităţile de I/O sunt folosite pentru a plasa date în memoria primară şi pentru a stoca cantităţi mari de date pentru o perioadă lungă de timp.
- Astfel, ele pot fi unităţi de stocare (unităţi bloc), cum ar fi discurile, respectiv unităţi caracter cum ar fi tastatura, mouse-ul, display-ul terminalului precum şi unităţi de comunicaţie, cum ar fi portul serial conectat la un modem sau o interfaţă la reţea.
- Fiecare unitate foloseşte un controller de unitate pentru a o conecta la adresele calculatorului şi la magistrala de date. Controller-ul oferă un set de componente fizice pe care instrucţiunile CPU le pot manipula pentru a efectua operaţii de I/O. Ca şi construcţie, controller-ele diferă, dar fiecare oferă aceeaşi interfaţă de bază.
- SO ascunde aceste detalii de funcţionare ale controller-lor, oferind programatorilor funcţii abstracte pentru accesul la o unitate, scrierea/citirea de informaţii etc.
- Controller-ul de unitate furnizează o interfaţă folosită de către mediul microprogramat de la cel mai înalt nivel. Componenta SO care manipulează dispozitivele de I/O este formată din driverele de unitate.

Moduri de lucru ale procesorului

- **Setarea modului de lucru**. Procesoarele contemporane conţin un bit care defineşte modul de lucru al procesorului. Acest bit poate fi setat în modul **utilizator** sau **supervizor.** În modul supervizor, procesorul poate executa orice instrucţiune cod maşină, pe când în modul utilizator el poate executa numai o parte dintre aceste instrucţiuni. Instrucţiunile care pot fi executate numai în modul supervizor se numesc instrucţiuni privilegiate.
- De exemplu, astfel de instrucţiuni sunt cele de I/O.
- Un proces, dacă este executat în mod utilizator, el nu poate să-şi execute propriile instrucţiuni de I/O. De aceea, aceste instrucţiuni sunt executate prin intermediul SO. Când un program de aplicaţie face o cerere către sistem, o instrucţiune cod maşină specială este apelată pentru a comuta procesorul în modul supervizor şi începe să execute driverul unităţii respectiv.
- Corespunzător celor două moduri de lucru, memoria internă este împărţită în zona de memorie utilizator şi zona de memorie supervizor. Dacă bitul mod de lucru este setat pe utilizator, atunci procesul respectiv are acces numai la zona cu acelaşi nume. Astfel, se realizează şi protecţia zonei de memorie supervizor.

Nucleul **SO**

- Este acea parte a SO care este executată în modul supervizor.
- Alte procese, legate de diverse aplicaţii ale utilizatorilor sau chiar aplicaţii soft de sistem sunt executate în mod utilizator.
- Execuţia acestor instrucţiuni nu afectează securitatea sistemului.
- Când un proces doreşte să execute anumite operaţii în mod supervizor, atunci se va face o **comutare** din modul utilizator, în cel supervizor.
- Acest lucru se realizează prin intermediul unei instrucţiuni **trap,** numită instrucţiune de apel al supervizorului, care setează bitul de mod de lucru şi face un salt la o locaţie de memorie, care se află în spaţiul de memorie protejat, locaţie care conţine începutul unei proceduri sistem care va rezolva cererea procesului.
- Când execuţia acestei rutine supervizor s-a terminat, SO va reseta bitul de mod de lucru din supervizor în utilizator.

Apelurile de sistem

- furnizează o interfaţă între un proces şi sistemul de operare. Prin intermediul acestora, un program utilizator comunică cu SO şi cere anumite servicii de la acesta.
- Apelurile de sistem pot fi împărţite în cinci categorii importante:

►**controlul proceselor**(încărcarea, execuţia, sfârşitul, abandonarea, setarea şi obţinerea atributelor, alocarea şi eliberarea de memorie etc);

►**manipularea fişierelor**(creere, ştergere, deschidere, închidere, citire, scriere, repoziţionare, setarea şi obţinerea atributelor);

► **manipularea unităţilor**(cerere şi eliberare unitate, citire scriere şi repoziţionare, obţinerea şi setarea atributelor, ataşarea/detaşarea logică);

► **întreţinerea informaţiilor(**obţinerea şi setarea timpilui sau datei calendaristice sau a sistemului, obţinerea de informaţii despre componentele fizice şi logice ale sistemului şi posibilitatea modificării lor**);**

► **comunicaţii**(crearea şi anularea unei conexiuni, transmiterea şi primirea de mesaje, transferul informaţiilor de stare, ataşarea/detaşarea logică a unităţilor la distanţă).

Limbajele de asamblare şi limbajele evoluate moderne conţin instrucţiuni (comenzi) prin care sunt lansate apeluri de sistem. După transmiterea parametrilor, este declanşată o instrucţiune trap, pentru oferi controlul SO. Când a terminat de executat rutina respectivă, SO returnează un cod de stare într-un registru, care specifică terminarea normală sau anormală şi execută o instrucţiune de revenire din instrucţiunea trap

Generarea, configurarea şi lansarea în execuţie a sist. de operare

- În cazul generaţiilor de calculatoare mai vechi, sistemul de operare este destinat numai unui anumit tip de calculator.
- **Exemplu.** Sistemul SIRIS este destinat familiei de calculatoare FELIX C-256,512,1024, iar sistemul de operare RSX este destinat minicalculatoatelor compatibile PDP, printre care se numără şi cele româneşti CORAL si IDEPENDENT.
- Sist. de op. moderne sunt realizate pentru a lucra pe o clasă de sisteme de calcul, care în multe cazuri diferă prin configuraţia lor. Sistemul trebuie să fie generat (sau configurat) pentru fiecare sistem de calcul în parte, proces care se numeşte **generarea sistemului**(SYSGEN). Programul SYSGEN citeşte dintr-un fişier sau cere informaţiile care se referă la configuraţia hardware a sistemului respectiv. Prin acest proces sunt determinate următoarele informaţii:
- ►Ce fel de CPU este folosită? Care dintre opţiunile(set de instrucţiuni cod maşină extins, instrucţiuni aritmetice în virgulă flotantă etc) sunt instalate? Dacă sistemul este multi-procesor, fiecare CPU va fi descrisă.
- ►Câtă memorie este disponibilă?
- ►Care unităţi sunt disponibile? Sistemul trebuie să cunoască adresele acestor unităţi, adresele întreruperilor corespunzătoare unităţilor, precum şi nişte caracteristici tehnice ale unităţilor.
- ►Setarea unor parametri ai sistemului de operare, cum ar fi de exemplu, numărul şi dimensiunea zonelor tampon utilizate, numărul maxim de procese care pot fi lansate la un moment dat, tipul algoritmului de planificare utilizat de CPU etc.

- După stabilirea acestor informaţii, sistemul de operare poate fi utilizat în mai multe moduri:
- administratorul de sistem poate modifica codul sursă al sistemului de operare, situaţie în care toate programele sursă vor fi recompilate şi vor fi refăcute legăturile;
- selectarea unor module noi care fac parte dintr-o bibliotecă precompilată, fiind necesară numai refacerea legăturilor;
- selectarea unor opţiuni se face în timpul execuţiei, situaţie caracteristică sistemelor de operare moderne.
- Unul dintre serviciile de bază ale unui **SO** este acela de a se putea **autoîncărca** de pe disc în memoria internă şi de a se autolansa în execuţie. Această acţiune se desfăşoară la fiecare punere sub tensiune a unui **SC**, precum şi atunci când utilizatorul doreşte să reîncarce **SO,** fiind cunoscută sub numele **încărcare sau lansare în execuţie** a **SO**. Pentru lansare se utilizează un mecanism combinat **hard** şi **soft**, numit **bootstrap**.

- Mecanismul **bootstrap** intră în lucru la apăsarea butonului de pornire <START> şi cuprinde următoarele etape:
- Se citeşte din memoria ROM un număr de locaţii consecutive. Aceste locaţii conţin instrucţiunile de copiere ale programului care va încărca nucleul SO. Motivul pentru care acest program este stocat în memoria ROM, este că aceasta nu trebuie să fie iniţializată şi că procesul respectiv se află la o adresă fixă, de unde poate fi lansat în cazul punerii sub tensiune sau re-setării sistemului. De asemenea, deoarece ROM este o memorie “read-only”, ea nu poate fi virusată.
- Se lanseză în execuţie programul citit la pasul anterior, care citeşte de pe un disc un program mai complex, care încarcă nucleul sistemului de operare.
- 3. Se lansează în execuţie programul citit la pasul anterior care va încărca nucleul sistemului de operare.
- **Observaţie.** Se pune firesc întrebarea, de ce programul care încarcă nucleul SO nu este citit din memoria ROM? Răspunsul este că instalarea unui alt sistem de operare, ar însemna modificarea acestui program, deci a conţinutului memoriei ROM, care este o problemă destul de costisitoare.

Tipuri de sisteme de operare

- **Sisteme de operare pentru calculatoare mari.** Calculatoarele mari sunt caracterizate de capacitatea de a procesa volume foarte mari de informatii. Ele lucrează cu dispozitive de I/O de capacitate mare de stocare şi sunt orientate spre execuţia mai multor sarcini în acelaşi timp. Pe astfel de calculatoare se pot executa: servere de Web, servere pentru site-uri dedicate comerţului electronic şi servere care gestionează tranzacţii între companii. Aceste sisteme de operare oferă trei tipuri de servicii: procesarea loturilor, procesarea tranzacţiilor şi partajarea timpului.
- Un sistem de procesare în loturi prelucrează lucrări obişnuite, care nu impun intervenţia utilizatorului.
- **Exemple** de astfel de prelucrări sunt: procesarea operaţiilor uzuale din cadrul companiilor de asigurări, raportarea vânzărilor dintr-un lanţ de magazine.
- Sistemele de procesare a tranzacţiilor gestionează pachete mari de cereri scurte.
- **Exemplu** de astfel de prelucrare este procesul de căutare/verificare într-o bază de date a unei companii aeriene care conţine rezervările de bilete.
- La un astfel de sistem de calcul, sunt legate terminale de la care se introduc un număr mare de tranzacţii diverse.
- **Exemplu** de astfel de sistem de operare este OS/370, care a fost realizat de firma IBM..

- **Sisteme de operare pentru calculatoare pe care se execută servere.** Aceste sisteme de calcul fac parte din reţele de calculatoare şi rolul lor este de a servi cererile aplicaţiilor lansate de diverşi utilizatori. În funcţie de complexitatea operaţiilor pe care le execută, aceste calculatoare pot fi de la calculatoare personale la calculatoare de putere foarte mare.
- **Exemple** de servere**:** Servere care administrează utilizatorii unei reţele de calculatoare, servere de fişiere, servere de baze de date, servere de poştă electronică, servere care asigură servicii de Web etc.
- **Sisteme de operare pentru calculatoare multiprocesor.** Aceste sisteme de calcul încorporează mai multe unităţi de prelucrare (procesoare), în vederea creşterii capacităţii de prelucrare. Aceste calculatoare necesită sisteme de operare speciale, care să rezolve diverse probleme specifice legate de încărcarea echilibrată a procesoarelor, partajarea resurselor etc.

- **Sisteme de operare pentru calculatoare personale.** Aceste sisteme de operare sunt concepute pentru a oferi posibilităţi de lucru unui singur utilizator. Ele pot funcţiona individual sau legate în reţea. În cadrul unei reţele, calculatoarele pe care se execută servere folosesc variante diferite de sisteme de operare faţă de cele ale sistemelor de calcul personale.
- **Exemplu.** Există patru versiuni ale sistemului Windows 2000: versiunea Profesional este destinată calculatoarelor individuale; celelalte trei versiuni (Server, Advanced Server şi DataCenter Server) sunt destinate calculatoarelor server dintr-o reţea. Windows 2000 DataCenter Server este destinat serverlor multi-procesor.
- Versiunea XP propune două variante: Windows XP. Net Server, utilizat de calculatoarele server dintr-o reţea şi Windows XP Proffesional, destinat calculatoarelor individuale precum şi calculatoarelor client legate în reţea.
- **Sisteme de operare pentru calculatoare în timp real.** Sistemele de calcul în timp real consideră timpul de răspuns ca o caracteristică fundamentală a bunei lor funcţionări.