1 Introducere

Ştiinţa tehnologiei informaţiei (Informatica) a apărut ca termen în urma dezvoltării mijloacelor electronice de prelucrare a informaţiilor, dar ca domeniu de activitate (şi de studiu) se poate spune că există din totdeauna – chiar dacă nu a avut în trecut acest nume. Diferite categorii de informaţii se prelucrează diferit dar, la nivel conceptual, se constată că modalităţile de prelucrare sunt asemănătoare, deci se pot aborda unitar. Tehnologia Informaţiei (în engleză „Information Technology" - IT) exprimă, aşadar, metode generale de lucru cu informaţia, şi astăzi a devenit o ştiinţă de sine stătătoare cu aplicabilitate în orice domeniu de activitate umană.

Domeniul informaticii este în ultimul timp legat de domeniul comunicaţiilor, atât datorită tehnicilor digitale folosite în ambele domenii cât şi datorită tendinţei de integrare a producerii, transportului şi consumului de informaţie. Astfel, a apărut „Tehnologia Informaţiei şi Comunicaţiilor" TIC (în engleză „Information and Communication Technologies" ICT), care unifică cele trei categorii de informaţii de bază (date, sunete şi imagini) şi le tratează unitar după digitizare (adică după transformarea din semnal în succesiuni de numere binare), iar prelucrarea se face cu acelaşi tip de echipamente (calculatoare numerice), folosind programe adecvate.

Tehnologia Informaţiilor şi Comunicaţiilor (TIC) este domeniul ştiinţific şi tehnic care se ocupă cu metodele şi procedeele de achiziţie, prelucrare, stocare, transport şi prezentare a informaţiilor, prin echipamente („hardware") şi aplicaţii („software") create în acest scop.

Utilizarea TIC presupune un set relativ restrâns de metode de uz general şi de „obişnuinţe" de operare (lucru cu calculatorul) pentru un registru larg de utilizatori, fiindcă chiar extinderea TIC a impus două caracteristici - ca cerinţe de bază:

- atributul *deschis* (în engleză „open") – prin care modificări în structura şi complexitatea unui sistem de calcul (ca echipamente şi programe) nu trebuie să afecteze modul de lucru al utilizatorului;
- atributul *prietenos* (în engleză „friendly") – prin care operarea la sistem nu trebuie să necesite cunoştinţe speciale (de exemplu nume de comenzi, sintaxă, etc.), manevre trebuie să fie simple, iar sistemul să ofere sugestii şi ajutor în lucrul utilizatorului.

În vederea lucrului comod şi uniform al utilizatorului în reţele de calculatoare eterogene (adică cele care interconectează echipamente şi programe de la diferiţi producători), s-a impus o a treia caracteristică deziderat, anume:

- *transparenţa* pentru utilizator a tipului de echipament sau program care oferă un anumit serviciu (de exemplu serviciul de poştă electronică).

Urmare a acestui atribut, utilizatorul nu trebuie să fie interesat de tipul şi locul echipamentului sau programului pentru consumarea unui serviciu ci să opereze la fel de oriunde în reţea. Astfel, au apărut standarde internaţionale care stabilesc modul cum trebuie să funcţioneze, şi chiar cum să se prezinte către utilizator, o aplicaţie de uz general (cum este de exemplu mesageria electronică).

1.1 Informaţii şi prelucrări

Pentru domeniul TIC informaţia este şi „materia primă" şi „produsul final". Informaţia nu poate fi separată de o utilitate, o finalitate umană, şi este în legătură cu procesul de cunoaştere. ***Informaţia*** este o piesă de cunoaştere legată de o utilitate umană, aşteptată şi/sau obţinută în urma unui eveniment, apoi stocată sau transmisă prin materie sau câmp.

De se vorbeşte însă, de o tehnologie a informației? După cum prelucrările pe care le suferă materiile prime într-o uzină spre a se obține un anume produs se bazează pe o tehnologie – ca succesiune bine stabilită de transformări efectuate cu un anumite instalații special proiectate, tot astfel, informația din mediu este prelucrată într-o succesiune bine definită de operații cu un echipament dedicat – calculatorul electronic. Pe această similitudine se poate concepe o paralelă între o tehnologie industrială şi tehnologia informațiilor, care poate fi sintetizată astfel: *materie primă – pregătire – transformare - ambalare* a produsului finit, respectiv *informație – reprezentare – prelucrare – prezentare* a rezultatelor către utilizator.

1.1.1 Date

În cazul său cel mai simplu, informația se referă la apariția (sau ne-apariția) unui eveniment, informația căpătată fiind de tip binar: Adevarat/Fals, Da/Nu. Efectiv, informația apare după consumarea evenimentului, adică atunci când piesa de cunoaştere este *sigură* şi *clară*; spunem că informația s-a instanțiat – în sensul că a căpătat o valoare pentru o mărime legată de eveniment. Omul manipulează însă şi informații care nu sunt sigure (sunt *probabile* sau sunt *posibile*) precum şi informații care nu sunt clare (sunt *vagi*). Totuşi, şi acestea pot fi prelucrate şi utilizate folosind tehnica de calcul, prin metode stochastice (probabilități) sau metode posibiliste, respectiv prin metode de inteligență artificială cum sunt tehnicile „Fuzzy”.

Date sunt informații într-o reprezentare adecvată stocării lor pe un suport (magnetic, optic, etc.), în scopul prelucrării electronice sau transferului electromagnetic.

Reprezentarea datelor în calculatoarele numerice se face prin *discretizare.* Acest procedeu indică o separare în piese (discrete) a elementelor ce compun informația. Toate informațiile în calculatoarele numerice sunt reprezentate prin coduri (numerice), pentru a fi ulterior stocate şi prelucrate după cerințe. Ca semnificație, discretizarea se referă la separarea unor momente de timp egale în evoluția unui fenomen real (şi care prezintă de fapt o curgere continuă), în scopul cunoaşterii sau modelării acestuia; prin extensie, se consideră discretizare şi descompunerea în piese spațiale a unui obiect (de exemplu o imagine).

1.1.2 Prelucrarea datelor

Funcție de scopul urmărit, datele suferă diverse transformări, mai mult sau mai puțin complexe, generic numite prelucrări (în engleză „data processing”); acestea nu sunt întotdeauna calcule ci, de exemplu, selectări ale unor piese de date din setul existent. În această idee, se pot deosebi următoarele prelucrări generice:

- ***Sortare*** – constă în *ordonarea pieselor de date* după valorile unei caracteristici comune acestora, numită *cheie de sortare*.
- ***Selectare*** – constă în *extragerea unor piese de date*, din setul existent, pe baza unui *criteriu de selecție*.
- ***Clasificare*** – constă în *împărțirea setului pieselor de date* în grupuri de date cu o *caracteristică comună*.
- ***Transformare*** – constă în *modificarea valorilor unor piese de date* prin operații în care intervin şi alte piese de date, constituind împreună *expresii*.
- ***Prezentare*** – constă în operații executate asupra pieselor de date doar în scopul *afişării adecvate* către utilizatorul uman a acestora.
- ***Transfer*** – constă în *transportul datelor între locații spațiale* diferite, prin infrastructura de comunicație, sub formă de unde electromagnetice.

Prelucrările se execută în paşi care se succed în secvență, cicluri sau ramificații, conform unor „rețete” numite *algoritmi*.

Algoritmul este o succesiune de operațiuni executate în număr finit de paşi, care prelucrează un număr finit de date inițiale (de intrare) către un număr finit de date rezultat (de ieşire).

Trebuie subliniat, însă, că există şi prelucrări electronice care nu se execută prin algoritmi. Acesta este cazul prelucrărilor prin „memorii asociative", cum sunt rețelele neuronale artificiale; acestea fac asocieri între perechi „dată de intrare – dată de ieşire", pentru *recunoaşterea formelor*, adică unei forme de intrare îi corespunde o formă de ieşire (de exemplu unei forme geometrice sau unei fizionomii umane va corespunde un nume). Simularea prin programe pe calculatorul numeric a acestor *memorii asociative* se face însă tot prin algoritmi, fiindcă acesta este singurul mod prin care un program poate fi executat pe calculatorul numeric.

Există şi prelucrări ne-algoritmice – de exemplu prin *rețelele neuronale artificiale*, care se realizează atunci când implementarea acestora se face fizic, în dispozitive şi echipamente dedicate. Se poate astfel ca, în viitor, calculatoarele de uz general să fie echipate cu o unitate pentru prelucrări algoritmice – folosind procesorul numeric, şi cu o unitate pentru prelucrări de tip recunoaştere – folosind procesorul RNA.

1.1.3 Scopul prelucrării electronice a informațiilor

Am văzut că informația vizează o utilitate umană, un scop; de aceea, este interesant să inventariem câteva utilități vizate astăzi de către prelucrarea electronică a informațiilor:

a) Crearea şi prezentarea de *documente scrise* de înaltă calitate (cu imagini, tabele şi formatări diverse), permițând modificarea rapidă şi eficientă a acestora prin operațiuni automatizate.
b) Realizarea de *calcule complexe* pentru aplicații inginereşti, economice sau ştiințifice, cu posibilități de interpretare şi generalizare a rezultatelor.
c) Realizarea de *prezentări către auditoriu*, cu imagini, sunete, text şi date, pentru informare, educație şi reclamă, accesibile local sau la distanță.
d) *Evidența şi controlul* operațiunilor economice, tehnice, spațiului geografic, transportului şi mediului social (folosind baze şi bănci de date).
e) *Conducerea proceselor tehnologice*, pentru asigurarea calității şi eficienței sistemelor de producție din industrie sau din servicii.
f) *Asistarea deciziilor* (adică oferirea de date sintetice şi sugestii) pentru conducerea (în sensul de management) a sistemelor economice, industriale sau sociale.
g) *Comunicare totală* prin intermediul rețelelor de calculatoare şi infrastructurii de comunicație, adică comunicare prin integrarea celor trei tipuri de informații uzuale: sunet, imagine, date. Comunicarea are loc între oameni, om-calculator sau între calculatoare.
h) *Divertisment* – ca jocuri pe calculator, discuții, filme, muzică, artă şi informare diversă.

Utilitățile oferite de aplicații uzuale astăzi, pe calculatoarele numerice de uz general, se referă la realizarea documentelor, calculelor tehnice sau economice, precum şi la comunicația inter-umană; acestea au în general o operare uniformă - indiferent de echipament sau producător, şi de aceea vor fi prezentate cu precădere în cartea de față.

Alte aplicații răspândite sunt cele de evidență şi control, dar acestea sunt dedicate unui domeniu de activitate dat, astfel că operarea este specifică domeniului şi chiar aplicației în cauză. Categorii de aplicații, cu detalii asupra lor, vor fi prezentate în capitolul 5.

1.1.4 Organizarea datelor

Informațiile sunt colectate de om şi stocate uzual pe diferite pe hârtie, în cărți, caiete, afişe. Pentru a fi utilizate ulterior, informațiile trebuie regăsite, iar pentru o regăsire rapidă ele trebuie să fie organizate, structurate. Este evident că mult mai greu se găseşte o carte aflată într-o grămadă, pe duşumea, față de cazul în care cărțile se află pe rafturi, aranjate pe domenii

sau autori indicate prin etichete. Din aceleaşi considerente de regăsire rapidă, datele (ca informaţii într-o reprezentare pe calculator) trebuie să fie organizate şi „etichetate” pe

suportul extern de memorare (disc magnetic sau optic).

Există trei modalităţi de organizare a unor piese de date, care se poate discuta privind eficienţa de regăsire a unei piese de date.

a) Informaţia este „nestructurată” – în care regăsirea unei piese dorite parcurge în cel mai rău caz tot setul de piese de mai multe ori.
b) În cazul în care informaţia este organizată într-o „structură înlănţuită” (*listă lineară*), regăsirea piesei dorite se face prin parcurgerea succesivă a pieselor existente, urmând sensul de înlănţuire a acestora. Regăsirea face, în cel mai rău caz, parcurgerea întregului set de piese o singură dată.
   În cazul în care informaţia este organizată într-o „structură arborescentă” (*listă neliniară*), deja piesele înlănţuite sunt şi clasificate pe niveluri, după un criteriu; în cazul cel mai defavorabil se execută un număr de paşi de căutare egal cu numărul de niveluri (minus unu). Un exemplu uzual al acestui mod de organizare a informaţiilor este un arbore genealogic de persoane, în care nivelul indică o generaţie iar relaţiile sunt legături părinte-fiu cu persoanele de pe nivel inferior.

Structura arborescentă este cea mai utilizată modalitate de organizare a datelor în Informatică, tocmai datorită eficienţei de regăsire a datelor. De fapt şi multe dintre cunoştinţele care vor fi prezentate în această carte sunt structurate arborescent, fiind şi pentru student organizarea care permite cel mai uşor înţelegerea şi memorarea cunoştinţelor.

1.1.5 Stocarea datelor (fişiere)

Pentru a fi stocate pe discuri, informaţiile cu semnificaţie sau utilizare comune se grupează în seturi, cărora li se ataşează o etichetă corespunzătoare. Dispozitivele de memorare „externă”, cum sunt discurile magnetice (fixe sau flexibile) precum şi discurile optice (cu înscriere unică sau multiplă) vor fi prezentate ulterior.

> ***Fişierul*** este o colecţie de date - de obicei de acelaşi tip, stocată pe suport extern.
>
> ***Directórul*** (dosarul) este o colecţie de fişiere - de obicei întrebuinţate în acelaşi context, stocată pe suport extern; un dosar poate conţine alte dosare.

Pentru organizarea eficientă a informaţiilor grupate în fişiere şi dosare, se utilizează - cum era de aşteptat, o structură arborescentă. În această structură, există un dosar – numit rădăcină, care conţine alte dosare (denumite în jargonul amintit „subdirectoare”), fiecare dintre acestea conţinând (sau nu) fişiere; cele care nu conţin direct fişiere conţin alte dosare care, la rândul lor, vor conţine fişiere Această organizare a dosarelor cu fişiere este denumită „structura arborescentă de directoare” a discului (magnetic sau optic) care stochează datele.

1.1.6 Introducerea şi prezentarea datelor

Introducerea şi prezentarea datelor constituie părţi foarte importante ale unei aplicaţii informatice, prelucrările fiind, spre exemplu, reduse la operaţii necesare stocării şi regăsirii datelor. Se poate spune că, la majoritatea aplicaţiilor actuale, achiziţia şi prezentarea datelor necesită 80% din volumul programelor realizate pentru acea aplicaţie.

Dispozitivele de introducere a datelor uzuale (*dispozitive de intrare*) sunt tastatura şi indicatorul („mouse”). Acestea sunt dispozitive relativ simple şi eficiente de comunicare om→calculator, dar care necesită un minimum de cunoştinţe privind convenţiile de lucru şi oarecare îndemânare pentru utilizarea lor. Dispozitive mai complexe – cum sunt cele pentru comandă vocală, nu sunt încă eficiente şi sunt rar utilizate datorită imperfecţiunilor inerente

tehnologiilor noi; avantajul lor este însă utilizarea conform uzanţelor umane comune, fără convenţii suplimentare.

Dispozitivele de prezentare a datelor uzuale (*dispozitive de ieşire*) sunt ecranul (în engleză „display") şi imprimanta (în engleză „printer"). Acestea se adresează vederii, prin care omul poate capta cel mai mare volum de informaţii, însă pentru înţelegerea semnificaţiei acestor informaţii este necesară cunoaşterea convenţiilor de reprezentare grafică, adică a simbolurilor afişate.

1.2 Comunicaţii de date şi multimedia

Termenul ***multimedia*** priveşte varietatea şi modalităţile de comunicare (imagine, sunet, date), prin care se combină canalele de comunicare umane. Aşa cum un bun profesor combină formulele înscrise pe tablă cu imagini, cu expresii verbale şi chiar cu tonalitatea vocii sale, spre a se face înţeles de clasă, unele aplicaţii prezintă informaţiile (în domenii ca cel comercial, de reclamă, de educaţie) atacând mai multe canale de cunoaştere: sunete, imagini (animate sau nu), texte, experimente filmate, etc. Informaţia este aşadar plasată pe mai multe medii, pentru receptarea ei de către auditoriu - de aici numele multimedia. Aceste tehnologii se referă atât la principiile şi metodele de prezentare cât şi la aparatura necesară pentru a obţine prezentări de înalt nivel.

Un alt termen, legat de comunicare este *hipermedia* (în engleză „hypermedia"), care se referă nu doar la diverse medii ce se adresează utilizatorului, şi totodată la accesarea de la distanţă a informaţiei, prin reţele de calculatoare.

1.3 Sisteme de calcul

TUtilizarea Tehnologiei Informaţiei şi Comunicaţiilor are cea mai largă extindere întâlnită la orice altă tehnologie cunoscută, datorită materiei prime pe care o prelucrează - informaţia. Cum nu există domeniu şi activitate umană care să nu folosească informaţii, TIC vine să sprijine activităţile din acel domeniu, prin metode şi mijloace pentru prelucrarea informaţiilor, unele generale altele specializate.

Se numeşte ***sistem de calcul*** complexul de *echipamente* fizice interconectate, cu *programele* şi *datele* legate intrinsec de funcţionarea şi utilizarea lor de către om.

Un sistem de calcul conţine „partea tare" – *hardware*, echipamentele, precum şi „partea moale" – *software*, programele şi datele.

Pe ansamblu, echipamentele unui sistem de calcul se împart în două categorii mari:

- *Unitatea centrală* – prelucrează datele şi stochează temporar valorile ce vor suferi prelucrări (date primare) şi de valorile rezultat.
- *Dispozitive periferice* – asigură introducerea, afişarea şi stocarea pe durată nedeterminată a datelor. Aceste dispozitive se mai numesc dispozitive de intrare/ieşire, pentru că asigură intrarea, respectiv ieşirea datelor din unitatea centrală pentru, respectiv după, realizarea prelucrărilor.

Unitatea centrală cuprinde *procesorul* (ca dispozitiv de prelucrarea a datelor) şi *memoria internă* (dispozitiv de stocare temporară a datelor), iar dispozitivele periferice uzuale sunt ecranul, indicatorul pe ecran („*mouse*"), discul optic (*CD-ROM*). Dispozitivele periferice sunt caracterizate drept dispozitive de intrare sau de ieşire, având drept referinţa memoria de lucru (RAM) a calculatorului.

Trebuie făcută o deosebire clară între *memoria internă* RAM şi dispozitivele periferice de memorare (cum sunt discul magnetic, discul optic, banda magnetică) – numite şi *memorii externe*. În timp ce informaţia stocată în memorii externe este păstrată un timp nedeterminat (oricât de lung şi fără a se pierde dacă echipamentul de calcul nu mai este alimentat cu energie electrică), informaţia stocată în memoria internă este păstrată doar pentru a fi prelucrată, dar se pierde la decuplarea alimentării calculatorului.

1.3.1.1 Procesorul

Prelucrarea datelor este efectuată de procesor care, uzual astăzi, este integrat într-un singur chip şi este numit microprocesor. Există două familii de microprocesoare:

- Familia Intel – folosite în calculatoarele „IBM compatibile" şi provin de la firma cu acelaşi nume; în această familie se pot încadra şi procesoarele produse de alte două firme: AMD şi Cyrix, ale căror produse sunt compatibile cu ale firmei Intel.
- Familia Motorola – folosite în calculatoarele produse de firma Apple.

Programele scrise pentru o familie de microprocesoare nu se pot „porta" (adică transfera şi rula imediat) pe altă familie.

Câteva caracteristici de performanţă a procesoarelor (actuale):

a. *Tipul procesorului* – se referă la produsul efectiv (producător, tip şi generaţie). PC-urile actuale au, spre exemplu, procesoare de tip Intel Pentium III şi IV sau AMD Athlon.
b. *Frecvenţa de lucru* (sau ceasul intern) – se măsoară în MHz (impulsuri de ceas pe secundă) şi stabileşte câte operaţii pe secundă poate efectua procesorul. Uzual, la PC-urile actuale, frecvenţa de lucru este cuprinsă între 1000 MHz şi 3000 MHz.
c. *Lungimea cuvântului procesor* – se măsoară în biţi şi indică numărul de biţi (sau octeţi) ce pot fi prelucraţi simultan de procesor. Procesoarele actuale au lungimea de cuvânt cuprinsă între 16 şi 64 biţi.
d. *Viteza procesorului* – se măsoară în „milioane de instrucţiuni pe secundă" (MIPS) sau (pentru cele foarte performante) în „milioane de operaţii cu virgulă mobilă pe secundă" (Mflops). Viteza procesorului este un parametru combinat, legat de frecvenţa de lucru, lungimea cuvântului de procesor şi tipul (producătorul şi versiunea) procesorului. PC-urile actuale au viteze cuprinse între 600 şi 1000 MIPS.
e. *Memoria cache* – cu capacitate indicată în multipli de Byte şi având rolul de memorie tampon pentru un lot de instrucţiuni preluate din memoria de lucru, spre a fi disponibile procesorului mai rapid decât accesate prin magistrală de la memoria de lucru. Există procesoare cu două memorii cache, pe nivel 1 şi 2.

Pentru staţii de lucru grafice şi servere cu prelucrări intensive, se folosesc două procesoare (în sistemele bi-procesor). Firma Intel oferă procesorul Xeon™, pentru aplicaţii „multitasking" (mai multe programe rulate simultan), lucrând la 3 GHz şi având memorie cache de 1MB.

1.3.1.2 Memoria internă

Programele şi datele se găsesc, pe durata prelucrării, în *memoria de lucru* (RAM), care este şi locul în care se introduc datele (de intrare) şi se pregătesc pentru afişare rezultatele (datele de ieşire). Performanţele memoriei interne se referă la tipul şi caracteristicile memoriei de lucru:

a. *Capacitatea de memorie* – indică volumul de date care se poate stoca în memoria de lucru şi se măsoară în multipli de octeţi (Bytes).
b. *Timpul de acces* – se referă la durata de timp scursă de la cererea unei date din memorie până la sosirea sa în procesor.
c. *Verificarea erorilor* – facilitate care indică dacă informaţia memorată este sau nu corectă,

1.3.1.3 Magistralele sistemului

Transferul datelor pe magistrale este sincronizat de un ceas al magistralei, care indică şi rata sa de transfer (debitul în bit/sec). Se prezintă pe scurt tipuri de magistrale cu caracteristici şi utilitatea lor:

a. *Magistrala ISA* (pe placa de bază) – magistrală provenită din generaţii mai vechi.
b. *Magistrala PCI* (pe placa de bază) – este magistrală mai rapidă, modernă.
c. *Magistrala AGP* (pe placa de bază) – oferă o rată mare de transfer (până la 21 GBps) la adaptorul de afişare grafică pe ecran (placa video), pentru prelucrări grafice 3D.
d. *Magistrala SCSI* – este o facilitate de transfer date către periferice rapide.
e. *Magistrala USB* – este o magistrală serială de mare viteză, pentru conectarea dispozitivelor periferice cascadate.

1.3.1.4 Placa de bază

Reprezintă suportul pentru dispozitivele electronice ce formează partea cea mai importantă a calculatorului: unitate centrală (cu microprocesor şi memorie de lucru), magistrale de comunicaţie cu perifericele şi adaptoare pentru periferice uzuale (tastatură, mouse, discuri, eventual reţea şi sunet).

1.3.1.5 Unităţi periferice de intrare

Datele pot fi introduse în memoria de lucru prin intervenţia omului (folosind tastatura, mouse-ul, scanner-ul, etc.) sau direct din mediu prin senzori şi dispozitive periferice dedicate. Perifericele de intrare uzuale sunt:

a. *Tastatura* – este dispozitivul de intrare cel mai utilizat şi prezintă butoane (taste) dedicate pentru simboluri alfa-numerice de scriere şi pentru comenzi de introducere a datelor (salt la nou rând, ştergere, etc.).
b. *Mouse*-ul – este dispozitivul de indicare a obiectelor grafice pe ecran, prin poziţia unui indicator de mouse (de obicei săgeată). Se pot executa manevre asupra obiectelor prin apăsarea simplă a butonului stânga, dreapta, dublu -click, glisare.
c. Folie sensibilă („*touch pad*”) şi bilă de urmărire („*trackball*”) – sunt dispozitive similare mouse-ului, folosite mai ales la calculatoare portabile („lap top”).
d. *Creion optic* – este un dispozitiv similar mouse-ului care permite indicarea directă pe ecranul (special) al monitorului; util pentru grafică.
e. *Joystick* – permite acţiuni de „navigare” pe ecran (similare tastelor săgeţi), precum şi acţiuni de comandă prin butoane speciale..
f. *Scanner* – este dispozitivul de introducere a imaginilor de pe suport hârtie. Performanţa acestuia este dată de densitatea de puncte pe ţol pe care le poate sesiza pe imaginea de intrare (dpi - dots per inch), cu valori de până la 2400×2400 dpi.
g. *Microfon* – este parte a complexului audio (placa audio şi dispozitivele exterioare), util la aplicaţii multimedia.
h. *Intrare video* – facilitate de preluare semnal complex de televiziune, pentru prelucrare şi memorare digitală.
i. *Intrări analogice* şi *intrări digitale* – ca denumiri generice pentru dispozitive care pot prelua din mediu semnale continui, respectiv discrete, utile în supravegherea şi conducerea proceselor industriale.

1.3.1.6 Unităţi periferice de ieşire

Pentru interacţiunea completă cu omul, după introducerea datelor sau chiar în timpul acesta, calculatorul prezintă pe dispozitivele de afişare situaţia curentă sau rezultatele prelucrărilor. Tipuri de periferice de ieşire uzuale sunt:

a. *Monitorul* (ecranul, „display") – afişează date sub formă de text şi imagine, prin puncte minuscule („pixeli") pe un ecran (fluorescent, cu cristale lichide sau cu plasmă).
b. *Imprimanta* – care permite imprimarea pe hârtie, folie transparentă sau hârtie specială (de exemplu hârtie fotografică) a textelor şi imaginilor; imprimante: *cu jet* (cerneala stropeşte prin mai multe ajutaje hârtia), *laser* (praful de toner se lipeşte termic, printr-un procedeu de impresionare a hârtiei similar celui de la copiatoare), *cu impact* (sau matriceale, în care un set de ace lovesc panglica cu cerneală în puncte verticale, deplasate apoi orizontal).
c. *Masă de desen* („plotter") – permite desenarea pe hârtie (de obicei de dimensiuni mai mari ca formatul A4) cu ajutorul unei peniţe purtate în cele două dimensiuni sau doar orizontal – pe verticală fiind antrenată hârtia.
d. *Difuzoare* – placa de sunet este adaptorul periferic pentru comunicaţii audio, ieşirea de sunet făcându-se pe difuzoare.
e. *Ieşiri analogice* şi *digitale* – similar intrărilor e acest fel, calculatoarele cu acţiune în procese industriale emit semnale electrice continui (analogice) şi digitale (discrete) pentru comanda echipamentelor de conducere şi control a instalaţiilor industriale.

1.3.1.7 Unităţi de memorare pe suport extern

Pentru stocarea pe o durată nedeterminată a datelor, se folosesc suporturi de memorare de tip magnetic sau optic. Dispozitivele care permit citirea (şi eventual scrierea) pe suportul extern de stocare se numesc *unităţile de memorie externă* („drives"); fiindcă suportul de memorie este de obicei sub formă de disc, uzual vorbind de unităţi de discuri, însă pot fi şi sub formă de panglică (bandă). Acestea au tipuri şi utilizări după cum urmează:

a. *Disc fix* („hard disk" HDD) – este unitatea de memorare necesară intrinsec funcţionării calculatorului, având stocate programele de bază şi aplicaţie precum şi datele de lucru cu acestea.
b. *Disc flexibil* („floppy disk" FDD) – este un suport amovibil (adică ce poate fi extras şi montat pe alte calculatoare), denumită uzual dischetă.
c. *Disc optic* („compact disc" CD) – este o memorie amovibilă care, în general, poate fi numai citită (numită de aceea CD-ROM), de mare capacitate (60 MB până la 80 MB).
d. *Bandă magnetică* („streamer") – este banda magnetică, de obicei bobinată pe role plasate într-o casetă de plastic, potrivite pentru salvare de siguranţă („back-up") pentru programe şi date ce se doresc păstrate un timp îndelungat.

1.3.1.8 Tipuri constructive de calculatoare

Cele mai utilizate calculatoarele sunt cele personale (PC „personal computer"), care prezintă, constructiv, variante legate de modul cum se aşează şi utilizează calculatorul:

a. Pe birou „*Desktop*" – cu forma carcasei plată (puţin înaltă), aşezată de obicei pe masa de lucru iar deasupra monitorul.
b. Lângă birou, *Turn* („*Tower*") – cu forma carcasei mai înaltă şi îngustă, aşezată de obicei lângă masa de lucru (pe podea sau în spaţiu special) iar monitorul pe birou.
c. Pe braţe („*Lap Top*" sau „*Notebook*") – cu formă foarte plată şi compactă, uşoare, cu tastatură şi ecran (cu cristale lichide - LCD) încorporate. Acest tip permite transportul comod şi lucrul în călătorii, fiind alimentat şi de la baterii.
d. În palmă *Palmtop* – foarte miniaturizate dar cu posibilităţi mai reduse de prelucrare decât precedentele. De fapt, cea mai mare parte a „calculatorului" o reprezintă ecranul, iar lucrul interactiv este realizat utilizând un creion electronic de indicare şi scriere.

1.3.2 Structura logică a unui sistemului de calcul

Din punctul de vedere al programatorului, sistemul de calcul are patru blocuri principale, utile în organizarea prelucrării datelor prin programul pe care îl va realiza.

Procesorul este responsabil de prelucrarea datelor şi controlul întregului calculator, unităţile de Intrare şi de Ieşire sunt responsabile de introducerea datelor şi prezentarea rezultatelor, iar Memoria este responsabilă de stocarea intermediară a datelor şi programelor. La momentul programării numai aceste patru blocuri logice sunt prezente în viziunea programatorului.

1.3.3 Principiul de lucru al unui sistem de calcul

Pentru simplitate, modul de lucru al calculatorului se poate asemăna cu modul de lucru uman - de exemplu în cazul rezolvării de către un student a unei probleme ce presupune calcule:

Prelucrările din aplicaţiile uzuale nu sunt totdeauna calcule (după cum s-a arătat mai sus), dar orice aplicaţie uzuală necesită operaţii de introducere a datelor şi de afişare a rezultatelor, fie direct fie prin intermediul unor module exterioare aplicaţiei de bază.

Student	Calculator
- citeşte datele de enunţ - aplică metoda de rezolvare: - realizează calcule, - reţine rezultate parţiale şi finale, - înscrie rezultatele pe hârtie (în caiet).	- *datele de intrare* introduse de la tastatură - se aplică *algoritmul* de prelucrare - date din memoria internă suferă operaţii prin intermediul procesorului, - rezultatele parţiale şi finale din procesor se transferă în memoria internă, - *datele de ieşire* se stochează pe disc sau afişează pe ecran, în formate prestabilite.

1.4 Rezumat

Informatica integrează astăzi tehnologia informaţiilor cu comunicaţiile şi are aplicaţii în orice domeniu de activitate umană. În acest scop, produsele informatice trebuie să poată fie utilizate de omul obişnuit, pentru prelucrări diverse, cum sunt prelucrări de birou, calcule ştiinţifice sau divertisment. Informaţiile sunt transformate în date prin reprezentări adecvate – ca numere binare, şi organizate în structuri pentru regăsirea rapidă. Echipamentele de calcul prezintă unităţi de de achiziţie, de prelucrare şi de prezentare a datelor. Memoria internă – pe lângă procesorul calculatorului, reprezintă partea care reţine datele de intrare, programul de prelucrare şi rezultatele. Memoriile externe au rolul unor depozite de stocare pe timp nedefinit a programelor şi datelor aferente, acestea fiind încărcate în memoria internă pe durata execuţiei programului ales la iniţiativa utilizatorului.

1.5 Teme de control

1. Ce scopuri se urmăresc prin reprezentarea informaţiilor ca date?
2. Daţi un exemplu de sortare pentru date personale înscrise într-un tabel de salariaţi.
3. Daţi un exemplu de clasificare pentru date privind situaţia şcolară a studenţilor.
4. Care este cea mai eficientă structură de organizare pentru regăsirea informaţiilor.
5. În ce mod este avantajat utilizatorul de produse informatice de atributele „deschis” şi „prietenos” al acestora?
6. Daţi exemple de dispozitive de ieşire a datelor, indicând o caracteristică de performanţă importantă.
7. Ce dispozitive de intrare şi ieşire a datelor sunt cel mai mult implicate în aplicaţii multimedia?

8. Ce rol are magistrala de pe placa de bază a unui PC?
9. Ce rol are memoria internă în prelucrarea datelor în calculatoarele numerice?
10. Ce asemănări şi ce deosebiri găsiți între fişier şi director (folder)?
11. Dați un exemplu de dispozitive periferice care sunt şi de intrare şi de ieşire.
12. Care dintre cele două tipuri principiale de memorie stochează temporar informația: memoria internă sau memoria externă?

Exemple de răspuns:

(3) Clasificarea este împărțirea unui set de date în grupuri cu caracteristici comune. Un exemplu de clasificare pentru situația şcolară poate fi gruparea studenților ca absolvenți, neprezentați şi picați - la un anume examen.

(10) Ambele sunt colecții şi sunt stocate pe disc; doar fişierul este colecție efectivă de date, în timp ce directorul este un catalog de fişiere util organizării informației de pe disc.

2 Reprezentarea şi structurarea informaţiei

Calculatoarele actuale sunt denumite *calculatoare numerice* (sau digitale – de la cuvântul englezesc „digit" - cifră) pentru că ele transformă toate informaţiile pe care le stochează şi prelucrează în reprezentări discrete. Au existat şi calculatoare analogice, în care informaţia păstra reprezentarea continuă, firească, a semnalelor din lumea înconjurătoare; aceste calculatoare prelucrau semnalele continui cu ajutorul „dispozitivelor electronice analogice" de tip amplificator de semnal, integrator cu condensator electric, dar limitările lor privind aplicaţiile şi modalitatea de stocare a datelor au dus la înlocuirea lor cu dispozitive numerice, care prin discretizare şi cuantificare pot reprezenta şi prelucra orice semnal analogic (cu o precizie mai mare sau mai mică).

2.1 Bit, octet şi multipli

Informaţiile logice, de tip Adevarat/Fals sau Da/Nu, sunt informaţii abstracte, putând fi considerate *piese elementare de informaţie* fiindcă au doar două valori posibile şi reprezintă situaţii simple sau generale – de exemplu un eveniment oarecare se produce (Da) sau nu se produce (Nu). Piesa de informaţie ce poate avea doar două valori posibile se numeşte ***bit***.
În calculatoarele numerice, un bit se consideră că poate lua ori valoarea 0 ori valoarea 1, fiecare din acestea corespunzând de fapt unei situaţii fizice în care un comutator este deschis – respectiv închis. Memoriile semiconductoare ale calculatoarelor numerice pot stoca biţi în miliarde de condensatoare minuscule, numărul acestor biţi reprezentând capacitatea de memorie a respectivului calculator. Asemenea numere uriaşe nu sunt comod de utilizat, de aceea se utilizează multipli de bit.

Octet (sau ***Byte***) este un grup de opt biţi reprezentând, tradiţional, o celulă de memorie.

Multiplii de bit şi Byte au denumiri similare celor uzuale la multiplii unităţilor de măsură, dar cu semnificaţie diferită: de exemplu, Kilo are semnificaţia 2^{10} şi magnitudinea 1.024, iar Mega are semnificaţia 2^{20} şi magnitudinea 1.048.576.

2.2 Identificator, variabilă, constantă, literal

Pentru a fi manipulate, datele trebuie referite, adică apelate după un nume sau prin adresa acestora în memorie. Fie prelucrarea algebrică următoare:

```
a = b + pi + 15 + sin(c);
```

unde `a,b,c,pi`, sunt date denumite prin litere, `sin()` este numele funcţiei trigonometrice sinus iar `15` este o valoare imediată.

Identificatorul este numele ataşat unei piese sau colecţii de date, dar şi a unei prelucrări anume, în general aflate în memoria de lucru.

Variabila este o locaţie de memorie referită printr-un identificator şi poate conţine diverse valori ale unei date de tip specificat.

Valorile pe care le poate lua variabila aparţin domeniului specific tipului de date respectiv. Datele vehiculate într-un program sunt stocate în variabile, care au primit fiecare o valoare chiar la momentul declaraţiei sau ca rezultat al unei expresii (prin atribuire).

Constanta este o locaţie de memorie referită printr-un identificator şi poate conţine doar o singură valoare a unei date de tip specificat.

Literal este o valoare concretă pentru un tip de date, indicată prin simboluri specifice tipului respectiv.

2.3 Tipuri de date simple

Varietatea informațiilor utilizate de om pare foarte mare, însă ele pot fi clasificate în categorii generice care apoi să primească reprezentări generale ca *tipuri de date*. Se poate face o analiză sumară din care rezultă câteva categorii de asemenea informații simple:

a) numere singulare – utilizate a desemna sume de bani, distanțe, zile din luna calendaristică, cu care se pot face calcule matematice. Se deosebesc două categorii de numere – *numere întregi* şi *numere reale* (cu zecimale).
b) simboluri grafice - litere, cifre, semne de punctuație, spații libere, utilizate în texte. Aceste simboluri sunt elemente ale unui set finit de simboluri de scriere (alfabetul, cifrele de la 0 la 9, semnele de punctuație); ele sunt denumite generic *caractere alfanumerice*.
c) Informații de tip Adevărat sau Fals – pentru a desemna o situație sau un eveniment funcție de care se ia o decizie (de exemplu, dacă un eveniment a avut loc se execută o anumită acțiune, altfel altă acțiune), denumite informații *logice*.

2.3.1 Reprezentarea numerelor întregi

Numerele utilizate curent pentru calcule (şi nu numai) se prezintă în aşa-numita „scriere arabă", în care cifrele sunt aranjate pe ranguri corespunzătoare puterilor lui 10, numărul 10 considerat „bază de numerație".

...	Mii	Sute	Zeci	Unități
...	10^3	10^2	10^1	10^0
...	5	9	9	9

$$5999_{(10)} = 9 \cdot 10^0 + 9 \cdot 10^1 + 9 \cdot 10^2 + 5 \cdot 10^3$$

Numărul 5999 în scrierea arabă, cu baza de numerație 10.

După cum se ştie, în reprezentarea zecimală a numerelor, rangurile conțin cifre care pot lua valori de la 0 la 9. Numărul de simboluri folosite pentru cifre este 10 (adică un număr egal cu baza de numerație).

2.3.1.1 Numere binare

Calculatoarele numerice sunt dispozitive realizate tehnologic în cel mai simplu mod, prin faptul că folosesc cea mai simplă reprezentare a informației: Adevărat / Fals sau Da / Nu. Aceste două simboluri pot fi cifre pentru numere în baza de numerație 2, fiindcă, similar lucrului în baza de numerație 10, un număr reprezentat în baza de numerație 2 va avea rangurile ca puteri ale bazei.

...	rang 3	rang 2	rang 1	rang 0
...	2^3	2^2	2^1	2^0
...	1	0	1	1

$$1011_{(2)} = 1 \cdot 2^0 + 1 \cdot 2^1 + 0 \cdot 2^2 + 1 \cdot 2^3 = 11_{(10)}$$

Numărul 11 în scrierea arabă, cu baza de numerație 2

Adunarea a două numere în baza 2 decurge după regulile cunoscute din baza 10, adică se adună cifrele pe ranguri şi, eventual, transportul de la rangul inferior, procedeul începând cu rangul cel mai puțin semnificativ. Din cele prezentate, rezultă că folosind doar două cifre (0 şi 1) se pot forma numere şi se poate opera cu ele în modul cunoscut. Aceste numere sunt denumite *numere binare* iar scrierea în baza 2 este denumită reprezentare binară a numerelor.

Numerele negative presupun existența semnului ce se ataşează numărului natural (pozitiv). Aşa cum numerele negative reprezintă o convenție din lumea matematicii (în lumea reală ele indicând o „lipsă"), la reprezentarea numerelor negative în calculator se face uz de o convenție prin care rangurilor disponibile pentru reprezentarea numărului se adaugă un rang suplimentar (cel mai semnificativ), indicând semnul:

- 0 – numărul este pozitiv (semnul +),
- 1 – numărul este negativ (semnul –).

2.3.1.2 Tipuri de date întregi

Limbajele de programare prevăd diferite categorii de numere întregi, clasificate după numărul de ranguri disponibile pentru reprezentarea numărului şi după prezența bitului de semn. Lungimea a rangurilor disponibile pentru reprezentarea numărului exprimă „precizia" acestuia şi este exprimată în biți sau octeți.

Se prezintă mai un jos exemplu de declarație pentru variabile de tip întreg în limbajul C, pentru variabile de tip întreg (pe doi octeți), indicând ziua din lună şi trei note de examen:

```
int ziua=13;
int NotaExamen1, NotaExamen2, NotaExamen3;
```

În cazul variabila `ziua` este inițializată la valoarea specificată după semnul =, adică această valoare este „încărcată" în locația de memorie a variabilei chiar la momentul declarației, deci imediat după rezervarea spațiului de memorie necesar (2 octeți). Valoarea ce apare după semnul = este un literal.

2.3.1.3 Operații cu numere întregi

Un tip anume de date permite un set bine definit de operații permise asupra variabilelor declarate cu acel tip. Operațiile sau operatorii se aplică diferențiat funcțiile de caracteristicile tipului de date, mai jos fiind inventariate operații pe cele trei categorii.

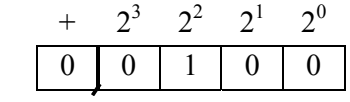
Asupra numerelor întregi se pot efectua următoarele *operații aritmetice* (în paranteze simbolul uzual al operației): adunare şi scădere (+ şi –), înmulțire (*), câtul împărțirii întregi (/), restul împărțirii întregi (%). Ultimele două operații sunt specifice numerelor întregi, rezultatul operației de împărțire fiind tot un întreg.

Numerele întregi pot fi comparate privind magnitudinea prin *operatori relaționali* cu simbolurile <, >, <=, >=; simbolurile <> sau != realizează comparația de neegalitate ≠, iar simbolurile = sau = = comparația de egalitate.

Tipul de date întreg este un *tip ordinal* cu ordine între valorile din domeniu: numărul 1 este înaintea numărului 2 şi după numărul 0. De aceea sunt posibile operații de tip predecesor – succesor; de exemplu operatorul de succesiune din expresia `succ(2)` are ca rezultat `3`.

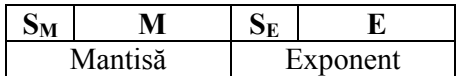
2.3.2 Reprezentarea numerelor reale

Pentru numerele reale care nu prezintă parte întreagă ci doar partea zecimală (adică numerele subunitare) se foloseşte o reprezentare binară imediată, în care virgula zecimală se consideră plasată între bitul de semn şi rangurile binare ale numărului propriu-zis.



↑virgula zecimală

a) număr real în virgulă fixă



b) număr real în virgulă mobilă

2.3.2.1 Operaţii cu numere reale

Fiind numere, cu numerele reale se pot efectua operaţiile aritmetice uzuale: adunare şi scădere (+ şi –), înmulţire (*),împărţirii cu zecimale (/). Între numerele reale se pot efectua comparaţii prin operatori relaţionali: <, >, <=, >=; comparaţia de neegalitate <> sau !=, apoi = sau = = pentru comparaţia de egalitate.

2.3.3 Tipul de date caracter

Simbolurile de scriere (fie ele în alfabetul latin, chirilic sau japonez) se numesc caractere (în engleză „characters") şi ele sunt necesare la scrierea textelor sau chiar a comenzilor prin care omul comunică cu calculatorul. Convenţiile de codificare sunt cuprinse în aşa-numitele „tabele de cod". Există mai multe asemenea tabele:

- tabela **ASCII** – pe 7 biţi, cea mai utilizată tabelă de cod (în calculatoarele personale şi alte sisteme de calcul), provenind din standardul american (American Standard Code for Information Interchange);
- tabela **Unicode** – pe 16 biţi, care este o altă extensie a tabelei ASCII (prima parte fiind identică cu tabela ASCII) şi care are un număr mare de coduri disponibile, pentru reprezentarea nu numai a caracterelor latine cu diverse diacritice dar şi a caracterelor diferite de cel latin (chirilic, arab, etc.).

2.3.3.1 Reprezentarea caracterelor

Fiecare caracter alfanumeric primişte un cod numeric conform tabelei de cod în care se reprezintă. Pentru a se deosebi de identificatori (care pot fi şi litere singulare), la manipularea caracterelor acestea se indică prin încadrarea simbolului de scriere între apostrof, adică se indică literali de tip caracter. Exemplu `'A'`, `'a'`, `'0'`, `':'`. Aceste simboluri nu au nici o semnificaţie în sistemul de programe, ci ele au rol numai de a fi desenate pe dispozitivele de ieşire (ecran, imprimantă) pentru a fi citite de om.

2.3.3.2 Operaţii cu tipul de date caracter

Tipul de date caracter este un tip ordinal, deci permite operaţii prin care se indică poziţia relativă a valorilor caracterelor, unele faţă de altele - predecesor `pred()`,succesor `succ()`.

Fiind reprezentate prin numere întregi, caracterele permit operaţii aritmetice admise pentru întregi. Între acestea, sunt utile cele de incrementare / decrementare (creştere / descreştere cu o unitate), prin care se poate parcurge tabela de cod sau şi se pot genera succesiuni de litere.

2.3.4 Tipul de date logic

În viaţa obişnuită sunt dese situaţii în care interesează apariţia unui eveniment; funcţie de apariţia acestuia pot apare alte evenimente sau se vor lua decizii în consecinţă. De exemplu, faptul că plouă ŞI fereastra este deschisă conduce la umezirea cărţilor de pe masa de lângă fereastră; dacă plouă, luăm decizia să închidem fereastra. Se poate, aşadar, ca intrând în camera cu pricina să constatăm că este „adevărat" că fereastra este deschisă, că plouă şi că se pot umezi cărţile. Cazurile de interes la apariţia unui eveniment sunt deci două - „adevărat" sau „fals", caracterizând din punct de vedere logic una din situaţiile de mai sus.

Datele care transpun în memoria calculatorului informaţiile despre situaţii caraterizate prin atributul „adevărat" sau „fals" se numesc date de tip logic, iar reprezentarea lor se face cu un singur bit: apariţia evenimentului este asociată valorii '1' iar ne-apariţia valorii '0' a bitului.

2.3.4.1 Operații cu tipul de date logic

Este posibilă combinarea mai multor situații logice (vezi exemplul de la începutul acestui paragraf), pentru cazurile când acestea:

- apar numai simultan (*conjuncția*) – ŞI logic (în engleză AND),
- apar alternativ sau simultan (*disjuncția*) – SAU logic (în engleză OR),
- apar în contradicție (*negația*) – NU logic (în engleză NOT),

care constituie chiar operatorii specifici ce se pot aplica datelor de tip logic.

Operatorii logici apar în programe, cel mai adesea, pentru a exprima condiții complexe în urma cărora se ramifică fluxul de comenzi ce se vor executa.

2.3.5 Tipuri de date structurate

Această categorie de date se referă la colecții de piese de date - simple sau structurate, numite şi *tipuri compuse* de date. Tipul fiecărei piese de date şi organizarea lor în structură se trebuie declarate la începutul programului, adică trebuie comunicate explicit compilatorului pentru a se face rezervarea corespunzătoare a locațiilor de memorie necesare. Structura astfel declarată nu se modifică pe parcursul programului, de aceea tipurile compuse sunt considerate *structuri statice* de date.

2.3.5.1 Tipul de date tablou

Deseori sunt necesare prelucrări asupra unui set de date de acelaşi tip; acestea se pot grupa în *masive* de date sau tablouri, cum sunt *vectorii* (tablouri cu o singură dimensiune) sau *matricele* (tablouri cu două sau mai multe dimensiuni).

Un ***tablou*** este o structură cu piese de date de acelaşi tip, fiecare piesă referită prin poziție.

Fiindcă datele de tip tablou sunt structuri ale altor tipuri de date, operațiile posibile asupra elementelor din tablou sunt cele permise pentru tipul de date respectiv.

2.3.5.2 Tipul de date şir de caractere

Simbolurile de scriere (caracterele) sunt utile mai ales pentru a se construi cu ele cuvinte, propoziții şi fraze, necesare comunicării inter-umane sau denumirii obiectelor (în interiorul sau exteriorul sistemului de calcul). Şirul de caractere este similar tipului de date tablou cu o singură dimensiune. Finalul de şir este marcat cu un cod special (`NULL`) iar la dispozitivul de afişare se transmit doar codurile caracterelor până la el, chiar dacă şirul rezervat este mai lung.

Operații asupra datelor de tip şir de caractere

Cu toate că reprezentarea şirurilor se bazează pe tipul de date tablou, operațiile posibile sunt specifice şirurilor de caractere şi sunt implementate prin *funcții în biblioteci* asociate limbajului de programare. Între acestea, cele mai uzuale sunt cele de „concatenare" a şirurilor (adăugare a unui şir după un altul), precum şi cele de înlocuire a unui subşir (parte a unui şir) cu un alt subşir, în scopul prelucrării textelor.

2.3.5.3 Tipul de date articol

Cele mai răspândite programe utilizate în domeniul economic (şi nu numai) sunt aplicațiile de gestiune. Acestea folosesc tabele care grupează datele referitoare la „obiecte" din lumea reală (cum sunt produsele dintr-un depozit, candidați la un examen, etc.), iar „gestiunea" constă în evidența şi prelucrarea datelor referitoare la acele obiecte. Un *tabel* se referă la o categorie de obiecte anume în care, la rândul lor, o *linie* se referă la un obiect anume iar o *coloană* (o rubrică) se referă la o anume proprietate sau atribut al acelui obiect. De exemplu, o linie din tabel se referă la un candidat anume iar coloanele conțin datele asupra sa (nume, prenume,

media, etc.). O linie din tabel se numeşte uzual *articol* (sau înregistrare – în engleză „record”) iar o celulă ce conține date (pe o coloană cu numele în capul de tabel) se numeşte *câmp* (în engleză „field”). Câmpurile pot conține date de tip şir de caractere (nume, prenume pentru un candidat), date de tip numeric (media), sau date de tip logic (a absolvit sau nu examenul).

Un ***articol*** este o structură cu piese de date de tipuri diferite, fiecare piesă referită prin nume. Prin intermediul articolelor se pot manipula datele din tabele, în memoria internă a calculatorului şi la transferul cu discul - memoria externă. Accesul al fiecare piesă de date (câmp) din articol se face prin numele rubricii (al câmpului).

Ca reprezentare a structurii de tip articol se poate imagina un tabel cu doar două linii: capul de tabel şi o linie cu date referitoare la un singur obiect - cel prelucrat la un moment dat.

Pentru prelucrarea pieselor de date din articol, acestea se vor referi prin numele asociat articolului şi numele asociat câmpului, respectând sintaxa:

```
nume_articol.nume_câmp
```

Similar tipului de date tablou, tipul articol este o structură ce conține alte tipuri de date, astfel că operațiile posibile asupra valorilor câmpurilor sunt cele permise pentru tipul de date din câmpul respectiv.

2.3.6 Tipuri abstracte de date – Clase de obiecte

După cum s-a constatat, la fiecare tip de date prezentate anterior, fiecare tip de date are specific un *nume*, o *semnificație*, un *domeniu de valori* şi *operații posibile* asupra acestor valori. În abordarea obiectuală, se definesc *clase* de obiecte, ca tipuri abstracte de date ce înglobează *structura de variabile* şi *operațiile* asupra lor. Inițial, clasa este descrisă privind datele (proprietățile) şi metodele (prelucrările) caracteristice obiectelor pe care le reprezintă.

Avantajul pe care programatorul îl are la folosirea claselor de obiecte este acela că poate declara orice structură de date corespunzătoare unei categorii de obiecte din lumea reală (o clasă de obiecte), defineşte operatori specifici clasei iar apoi foloseşte obiecte din această clasă în expresii, ca şi cum ar fi piese simple de date. Mai mult, chiar faza de proiectare a programului, se simplifică fiindcă nu vor trebui decât „imaginate” obiectele şi prelucrările necesare cu acestea, în scopul obținerii unui rezultat dorit. Spre exemplu, dacă se defineşte o clasă „paralelogram” – cu două operații posibile „modificarea dimensiunii a două laturi paralele” şi „modificarea unui unghi de colț”, se pot face prelucrări complexe cum sunt: transformarea paralelogramului în dreptunghi sau transformarea dreptunghiului în pătrat, folosind operatori şi simboluri definite specific în cadrul clasei. Valorile obținute după prelucrări, în acest exemplu, vor fi „dreptunghi” respectiv „pătrat”.

2.4 Rezumat

Capitolul descrie tipuri de date simple (cu câte o singură piesă de informație) şi tipuri de date structurate (cu mai multe piese de informație) şi reprezentarea lor în calculatorele numerice. Tipurile de date simple cuprind numerele (întregi şi reale), caracterele alfanumerice şi datele logice, iar tipurile de date structurate: tabourile, şirurile de caractere şi articolele. Reprezentarea în calculator a fiecăruia dintre acestea se bazează pe reprezentarea binară a numerelor întregi-pozitive, celelalte tipuri de date fiind ori codificări ori convenții de alăturare a unor biți sau alte piese de date suplimentare cu semnificație corespunzătoare. Tipurile de date abstracte realizează o generalizare a datelor şi operațiilor posibile asupra lor, încapsulate ca obiecte şi reunite în clase de obiecte; ele pot fi definite şi apoi utilizate în modul în care omul utilizează obiectele comune în viața reală.

2.5 Teme de control

1. De ce bitul este considerat piesă elementară de informație? Cum poate fi stocat bitul în calculatoarele numerice?
2. Ce are specific variabila şi prin ce este ea referită?
3. Ce înseamnă baza 2 şi reprezentarea binară a numerelor?
4. Câte ranguri sunt necesare pentru scrierea numărului 4 în baza 2; argumentați.
5. Indicați două operații cu numere întregi şi zecimale (cel puțin una diferită între ele), şi două operații relaționale cu simbolurile lor.
6. Ce este codul numeric al unui caracter alfanumeric şi cum se numesc inventarele de coduri?
7. Există operații cu date de tip caracter alfanumeric? Explicați.
8. Indicați mulțimea de valori şi mulțimea de operații de bază pentru date de tip logic.
9. Prin ce se deosebesc structurile de date de tip tablou şi de tip articol?
10. Un şir de caractere este asemănător unui tip de date tablou sau unui tip de date articol?
11. Ce este un câmp la tipul de date articol şi cum este el referit?
12. Indicați două avantaje în utilizarea obiectelor.

Exemple de răspuns:

(4) Pentru reprezentarea numărului 4 în baza 2 sunt necesare 3 ranguri; rangul al treilea indică puterea 2 a bazei – astfel că $2^2 = 4$, deci 3 este număr minim de ranguri.

(10) Un şir de caractere este asemănător unui tip de date tablou, pentru că în structura sa are mai multe piese de date de acelaşi tip (caractere alfanumerice) - ca şi tabloul, şi nu de mai multe tipuri – ca articolul.

3 Prelucrarea informației

În acest capitol se vor trece în revistă noțiuni privind modul în care se specifică diverse operațiuni efectuate asupra datelor şi modul în care se specifică desfăşurarea acestor operațiuni – conform prelucrărilor vizate. Abordarea sistematică a operațiunilor şi desfăşurării lor este impusă de exprimarea concisă şi completă a acestora printr-un limbaj de programare; fiindcă limbajele de programare moderne reprezintă de fapt codificări optime ale operațiunilor necesare în oricăror categorii de prelucrări.

3.1 Expresii

Numele calculator duce gândul la calcule – ca fiind prelucrările pe care acest instrument le poate executa dar posibilitățile de prelucrare ale calculatoarelor sunt mult mai largi.

Expresia este descrierea formală a unui set de acțiuni efectuate cu un anumit tip de date.

Semnul = are rolul de a „încărca" în variabila din stânga sa, valoarea rezultatului obținut în dreapta sa. Astfel, semnul = poate fi privit ca un *operator de atribuire*, iar întreaga construcție $F = m \cdot a$ ca o expresie cu o singură valoare – anume valoarea obținută pentru forța F.

Operatorii sunt simboluri ale unor acțiuni cu semnificație stabilită prin tipul de date operat. Am folosit deja simbolul `+` ca operator de concatenare (deci simbol al operației de concatenare). Operatorul poate fi format din unul sau mai multe caractere. Entitatea asupra căreia se aplică operatorul se numeşte *operand.* După numărul de operanzi, operatorii pot fi:

- *operatori unari,* care se aplica unui singur operand, de ex. operatorul - in expresia `-x`;
- *operatori binari,* care se aplică asupra a doi operanzi, fiind situat între aceştia; de exemplu operatorul de concatenare + in expresia `"acesta este"+" un exemplu"` sau operatorul de adunare a doua numere + in expresia `17+28`.
- *operatori ternari,* care se aplica asupra a trei operanzi; de exemplu în limbajele C şi Java există operatorul ternar `(? : )` folosit în expresiile condiționale.

Operatori aritmetici sunt operatorii prezentați mI sus. Există o ordine de precedență a operatorilor aritmetici: primii se aplică operatorii unari apoi cei de înmulțire şi împărțire, ultimii de sumare algebrică.

Operatori relaționali sunt operatori (binari) pentru comparație sau incluziune: <, >, = =, != (simbol pentru ≠), <= şi >= (simboluri pentru ≤ şi ≥), `in` (pentru apartenență∈).

Operatori logici sunt operatorii prezentați la §2.3.4.1 şi produc un rezultat logic („adevărat " sau „fals"). Ordinea de precedență este: primul negarea ! (NU), apoi conjuncția && (ŞI), ultimul disjuncția || (SAU); ordinea de precedență se poate schimba prin grupare cu paranteze.

3.2 Instrucțiuni

Pe lângă prelucrările variabilelor (care se realizează prin expresii), mai sunt necesare două tipuri generice de acțiuni: de decizie şi de repetiție. Rezultă că un limbaj de programare trebuie să asigure comenzi elementare (denumite instrucțiuni) pentru:

a) **Atribuire** – pentru de obținere a unui rezultat pentru o expresie şi de încărcare a valorii acestui rezultat în variabila țintă (din stânga semnului =).

b) **Decizie** – pentru alegerea unei secvențe de comenzi din mai multe variante (uzual două) urmare a rezultatului unei expresii logice – condiția de decizie.

c) **Repetiție** – pentru reluare a unei secvențe de comenzi de mai multe ori, până la îndeplinirea unei condiții de oprire.

Între aceste acțiuni doar atribuirea este cea care privește formularea expresiilor, rezolvarea și păstrarea rezultatelor. Decizia și repetiția modifică doar cursul de execuție al atribuirilor. Tabloul comenzilor posibile se completează cu:

d) **Salturi și reveniri** – pentru întreruperea unei secvențe de comenzi și preluarea controlului de către altă secvență de comenzi în cadrul aceluiași bloc program sau într-un subprogram.

e) **Operații de intrare / ieșire** – pentru interacțiunea cu exteriorul, prin periferice de intrare / ieșire (către om sau instalații, pentru stocarea sau transferul datelor).

Un **program** este un text ce cuprinde o secvență de instrucțiuni din categoriile de mai sus, care sunt executate în ordinea în care apar în text. Procesorul asigură preluarea și executarea comenzilor în secvență, aceasta fiind ideea centrală de funcționare a mașinilor von Neumann.

3.2.1 Instrucțiuni simple

Într-un program acțiunile sunt descrise prin linii de text, care apoi pot fi interpretate și executate de către calculator – linie cu linie sau pe loturi de linii. Descrierea unei acțiuni complete se încheie cu `;` ca simbol uzual indicând „sfârșitul acțiunii”. Asemenea instrucțiuni se prezintă în continuare.

3.2.1.1 Instrucțiunea de atribuire

Prelucrarea efectivă a datelor are loc în instrucțiunea de atribuire: una sau mai multe valori intră într-o *expresie* al cărei rezultat se atribuie *variabilei stânga* simbolului de atribuire = (în limbajele C și Java) sau `:=` (în limbajul Pascal). Pentru exemplul de calcul al forței:

```
F=m*a;    // în C, respectiv
F:=m*a;  /* în Pascal
```

3.2.1.2 Instrucțiuni de salt

Întreruperea forțată a executării unei secvențe de instrucțiuni (și saltul la începutul altei secvențe) se poate face condiționat (dacă a fost îndeplinită o condiție logică) sau necondiționat (neutilizat uzual), și instrucțiuni de salt care provoacă părăsirea secvenței de comenzi doar în condiții bine precizate, impuse direct de prelucrări.

Saltul de revenire din subprogram la programul apelant:

```
return [expresie];
```

unde `expresie` produce ca rezultat o valoare ce este „întoarsă” (returnată) programului apelant de către subprogram (vezi §3.2.3), fiind folosită direct în expresii.

3.2.2 Instrucțiuni structurate

Controlul fluxului de instrucțiuni este necesar pentru soluționarea unei probleme complexe, iar algoritmul este de fapt o rețetă de control al fluxului de instrucțiuni prin care se caută soluția la problema dată.

Omul execută operațiunile înscrise mai sus una după alta, fără a fi necesară indicarea ordinii lor prin numere Similar, procesorul (construit după principiile enunțate de von Neumann) execută operațiile una după alta, în ordinea apariției lor în textul programului.

3.2.2.1 Instrucțiunea de decizie binară

Pentru indicarea a două alternative în desfăşurarea acțiunilor unui program se foloseşte decizia sau ramificația binară „dacă .. atunci”, a cărei format general (în limbajul C) este:

```
if (expresie)
    instrucțiune1;
else
    instrucțiune2;
```

unde `instrucțiune1` se execută atunci când condiția `expresie` are valoare logică „adevărat” (sau în limbajul C, are o valoare diferită de 0), altfel (când condiția este falsă sau 0) se execută `instrucțiune2`. În cazul în care nu există o `instrucțiune2`, atunci ramura `else` lipseşte şi se continuă cu secvența ce urmează după simbolul de sfârşit `;`.

3.2.2.2 Instrucțiunea de decizie multiplă

Decizia binară – prezentată mai sus, priveşte situații simple, cu două alternative: Alb/Negru, Da/Nu, Adevărat/Fals. Pentru situații în care decizia priveşte mai mult de două alternative, este dificil de aplicat mai multe instrucțiuni de decizie binară. O asemenea situație apare când expresia de selecție nu are valori binare ci multiple – cum ar fi cazul selecției opțiunilor unui meniu; fiecare opțiune devine un caz selectat printr-un număr sau prin poziția indicatorului pe ecran („mouse”).

3.2.2.3 Instrucțiuni de repetiție

Deseori, prelucrările pentru care este util calculatorul sunt cele în care se repetă anumite operații de foarte multe ori; omul ar obosi (apoi greşi) la repetiții îndelungate ale aceloraşi operații, dar echipamentul electronic le execută precis, rapid şi fără complicații sociale.

Atunci când se cunoaşte numărul de iterații (cuvânt ce indică „repetiții numerotate”), este utilă instrucțiunea `for` – ce apare în diverse limbaje, iar în limbajul C ea este:

```
for (NrStud=1,NrBilet=35;NrStud<=30;NrStud++,NrBilet--)
    { „Prezintă legitimație şi primeşte bilet de examen” };
```

Exemplul priveşte verificarea legitimației de student şi primirea de către acesta a biletului de examen, la o grupă de 30 studenți. Contorul `NrStud` reține al câtelea student a intrat în sala de examen iar contorul `NrBilet` reține câte bilete au mai rămas examinatorului – din totalul de 35; expresiile `NrStud++` şi `NrBilet--` adună şi respectiv scad o unitate din contoarele respective (aceasta înseamnă `++` şi respectiv `--`).

3.2.2.4 Instrucțiunea de repetiție după condiție

Atunci când nu se cunoaşte numărul de iterații, terminarea repetițiilor poate fi indicată de o condiție cunoscută – care se aplică înainte sau după efectuarea operației în bucla de repetiție. Ca urmare, se folosesc una din cele două tipuri de instrucțiuni:

a) „cât timp (condiție) adevărată - repetă {operație}” – prin care este posibil ca operația să nu se execute nici măcar o dată, dacă de la început condiția nu este îndeplinită;
b) „repetă {operație} - cât timp (condiție) adevărată” – prin care operația se execută cel puțin o dată, indiferent dacă este adevărată condiția la intrarea în buclă.

(a)
```
while (NrStud>0)
        { „Prezintă legitimație şi primeşte bilet de examen” };
```

(b)
```
do
        { „Prezintă legitimație şi primeşte bilet de examen” };
while (NrStud>0);
```

3.2.3 Programe şi subprograme

În cazul programelor scrise într-un limbaj de programare, structura textului depinde de modul de programare (structurată sau obiectuală), în principiu, pentru un program fiind specificate:

```
nume_program (lista parametri)
{
     declaratii variabile
     corpul programului
}
```

unde `nume_program` este un identificator al programului (prin care poate fi apelat spre a executa acțiunile înscrise în el), `lista parametri` este setul de date care se furnizează programului (ca „materie primă") şi asupra cărora se vor executa acțiunile, `declaratii variabile` indică variabilele necesare stocării rezultatelor intermediare, `corpul programului` este secvența efectivă de comenzi pentru acțiunile vizate, iar acoladele `{}` încadrează şi delimitează programul propriu-zis.

3.2.3.1 Subprograme

În cazul unor prelucrări complexe, care sunt necesare repetat, este indicată folosirea subprogramelor; acestea sunt secțiuni de cod înscrie o dată şi folosite de mai multe ori, prin apelarea lor ori de câte ori este nevoie, separat sau în cadrul expresiilor. Apelarea subprogramului se face prin intermediul identificatorului său (numele), similar cu referirea unei variabile.

3.2.3.2 Programul principal

Execuția acțiunilor înscrise într-un program trebuie inițiată la comanda utilizatorului. Pentru aceasta sistemul de operare interacționează cu omul, primeşte comanda (care de obicei este chiar numele programului) şi „lansează" execuția acestuia.

Partea din textul unui program care poate fi lansată nemijlocit de sistemul de operare (deci care poate funcționa de sine stătător) se numeşte program principal. Acesta este, în general, o succesiune de acțiuni grupate în corpul programului pe trei secțiuni:

```
Introducerea datelor
Prelucrarea datelor
Afişarea rezultatelor
```

Pentru execuția acestora, se face apel la subprograme din biblioteca de subprograme sau din setul subprogramelor declarate în textul sursă al programului ca ansamblu; în acest ultim caz, programul va fi specificat astfel:

3.3 Algoritmi

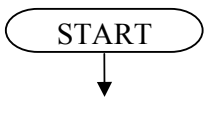
Algoritmii sunt modalități prin care se exprimă succesiunea de operații prin care un program pe calculator poate ajunge la datele de intrare furnizate la rezultatele dorite. Un algoritm nu exprimă doar operații cu numere ci orice fel de prelucrări (cu texte, imagini).

3.3.1 Exprimarea algoritmilor

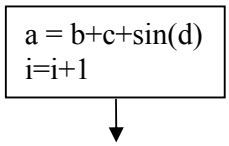
În general, descrierea prelucrărilor prin care se obține soluția unei probleme date se face fără a se specifica tipurile de date utilizate, ca şi cum acestea ar fi subînțelese. De aceea, exprimarea algoritmilor este o înşiruire de operațiuni în care se utilizează variabile şi (eventual) alte prelucrări, indicate prin identificatori inventariați într-un dicționar sau *nomenclator* ce specifică rolul variabilelor sau prelucrărilor respective.

3.3.1.1 Organigrame (Scheme logice)

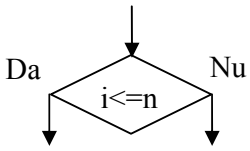
Succesiunea de operaţiuni ale unei prelucrări se poate descrie grafic prin blocuri standard. Chiar dacă are mai mult o valoare istorică, organigrama este un bun start în specificarea unui algoritm, fiind intuitivă şi uşor de constituit.



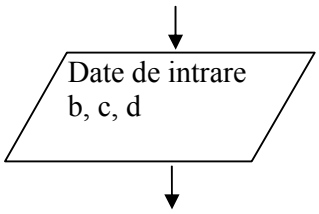
(a) Terminator iniţial



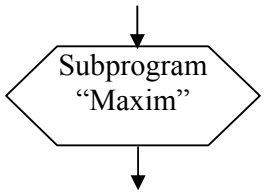
(b) Bloc de



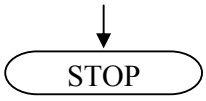
(c) Bloc de decizie binară



(d) Bloc de intrare/ieşire



(e) Apel de subprogram



(f) Terminator final

Succesiunea operaţiilor se indică prin linii şi săgeţi care leagă blocurile grafice prezentate, astfel constituind aşa-numita organigramă. Organigramele (scheme logice) nu mai sunt foloiste astăzi pentru descrierea programelor simple ci pentru structurarea din module a aplicaţiilor complexe. Mai mult, aceste organigrame se pot realiza asistat de un calculator (adică de un program de ajutor/asistare dedicate domeniului) după care calculatoru generează automat programul propriu-zis, în limbajul de programare dorit.

3.3.1.2 Pseudocod

Atunci când se doreşte nu doar descrierea algoritmului ci şi structurarea etapelor de rezolvare a problemei (prin modularizare), este indicată utilizarea unui limbaj codificat care exprimă (în limba maternă) operaţiunile de executat şi cele de control al fluxului de comenzi (cum sunt decizia binară, repetiţia). Modularizarea (adică separarea operaţiunilor pe secţiuni, fiecare cu un scop restrâns şi specific) este singura modalitate de abordare a rezolvării problemelor complexe şi cu soluţie puţin sau deloc cunoscută .

3.3.2 Elaborarea algoritmilor

Rezolvarea diferitor probleme din lumea reală cu ajutorul calculatorului se poate face doar dacă soluţia se cunoaşte dar trebuie căutată într-un set existent sau dacă există o metodă de găsire a soluţiei. Algoritmul este o asemenea metodă, dar elaborarea sa este de cele mai multe ori dificilă, considerată uneori (îndeosebi la începuturile programării) o artă.

Între metodele de elaborare a algoritmilor se amintesc cele mai importante (numele fiind indicat în engleză spre a fi uşor de recunoscut), cu o scurtă descriere a specificului lor:

- *Greedy* – pentru crearea de submulţimi optimale cu elementele preluate dintr-o mulţime dată şi cu respectarea unor restricţii impuse individual elementelor.
- *Backtracking* – pentru crearea de submulţimi optimale cu elementele preluate dintr-o mulţime dată, cu respectarea unor restricţii impuse elementelor dar şi setului (există relaţii între elementele submulţimii soluţie).

- *Divide et impera* – pentru probleme ce conțin secvențe sau piese discrete ordonate, care pot fi divizate în subprobleme care se rezolvă separat şi apoi se obține soluția prin combinarea soluțiile parțiale.
- *Branch and Bound* – pentru probleme în care soluțiile se pot reprezenta ca noduri în arbore iar căutarea soluției se face prin parcurgerea arborelui urmărind totodată o funcție de cost pentru a limita adâncimea de căutare.

3.4 Categorii de prelucrare şi prezentare a informațiilor

În acest paragraf se vor trece în revistă o serie de prelucrări uzuale realizate de sisteme de calcul, cu caracteristicile lor. Scopul paragrafului este de a se completa noțiunile prezentate în paragrafele anterioare, cu locul şi rolul prelucrărilor de date şi modului cum acestea sunt utilizate de către om.

3.4.1 Calcule matematice

Prelucrările matematice se întâlnesc în orice aplicație, fiindcă în orice domeniu al realității de utilizează mărimi cantitative. Tocmai de aceea, în general, limbajele de programare oferă – pe lângă operațiile generale de atribuire, decizie şi repetiție (denumite instrucțiuni de programare), un set de funcții pentru prelucrări matematice uzuale (cum sunt funcțiile trigonometrice, cele de aflare a părții întregi sau fracționare a numerelor reale) – grupate în biblioteci de funcții matematice.

Pentru efectuarea de calcule foarte laborioase – cum sunt calcule pentru modelarea fenomenelor în mecanica fluidelor sau pentru volume foarte mari de date (în economie, administrație, transport), precum şi pentru calcule a căror rezultat este aşteptat imediat („cu timp mic de răspuns"), se folosesc pentru prelucrarea datelor *algoritmi paraleli*.

3.4.2 Prelucrări de birou

În această categorie se pot încadra toate prelucrările care susțin omul în prelucrarea informațiilor care, în mod tradițional, foloseau ca suport hârtia sau mijloacele de comunicație de tip telefon sau poştă. Aceste prelucrări şi aplicațiile care le oferă privesc *utilizatorul obişnuit*, prin acesta înțelegând orice persoană care are de prelucrat documente „la masa sa de lucru" – fie acesta un funcționar, om de ştiință sau casnic:

- Prelucrări de documente „scrise" – folosesc *procesoare de texte*, ce oferă utilizatorului instrumente de editare, formatare şi corectură automată a textului şi altor elemente de exprimare a informațiilor (tabele, imagini).
- Calcule diverse şi ilustrarea seriilor cantitative prin histograme – folosesc *foi de calcul electronice*, ce permit utilizatorului calcule rapide pe volume relativ mari de date şi într-o modalitate simplă şi eficientă (fără a scrie programe speciale pentru acestea).
- Pregătirea şi expunerea materialelor de prezentare – folosesc *aplicații de prezentare către auditoriu*, ce permit expuneri multimedia către grupuri de oameni (folii de prezentare, sunet şi imagini animate).
- Gestiune şi prelucrare a datelor structurate (de tip articol) – folosesc *aplicații de baze de date*, ce permit crearea de tabele cu informații de diverse tipuri şi manipularea sau prelucrarea acestora.
- Comunicații în interiorul şi exteriorul organizației – folosesc *aplicații de transfer date* în intranet (de la şi spre colegi) sau în Internet (de la şi spre lumea largă).

Prelucrările din această categorie sunt foarte variate, aproape în majoritate fiind prelucrări algoritmice deterministe. Pentru faptul că eficiența şi aplicabilitatea lor devine tot mai largă,

tehnicile de Inteligență Artificială pătrund în aplicațiile obișnuite, deci astăzi ele folosesc şi algoritmi nedeterminişti (rețele neuronale artificiale, logică fuzzy, algoritmi genetici).

3.4.3 Prelucrări prin metode de Inteligență Artificială (IA)

În situațiile din viața curentă, omul trebuie să le rezolve mai rar probleme cantitative şi precise; adesea sunt de rezolvat probleme calitative într-un mod aproximativ. De exemplu, la traversarea unei străzi, în plin trafic, un om nu face calculul vitezelor maşinilor, a distanțelor şi stării drumului (etc., etc.) spre a aprecia apoi viteza cu care se va deplasa pentru a nu intra în coliziuni, ci procedează la un raționament aproximativ, bazat pe informații puține, dar eficient: traversează fără nici o zgârietură. De asemenea, nu efectuează calcule spre a recunoaşte o altă persoană sau o marcă de maşină ci, pe baza caracteristicilor lor cunoscute şi sesizate, decide spontan cine sau ce este.

Domeniul care se ocupă cu simularea comportamentului ființelor vii se numeşte *Inteligență Artificială* (IA). Implementarea tehnicilor IA se poate face fizic (prin sisteme electronice) sau logic (prin programe pe sisteme de calcul). Sunt vizate, în principal, tehnici pentru:

(1) Emularea raționamentului simbolic – prin *Sisteme Bazate pe Cunoştințe*, în care se parcurge un set de cunoştințe (exprimate ca propoziții) spre a obține noi cunoştințe.
(2) Reprezentarea şi inferența pentru informații aproximative – prin *Logică Vagă (Fuzzy)*, în care exprimări lingvistice aproximative (relative la cantități) intră în reguli „dacă .. atunci", spre a obține noi valori – aproximative sau precise, necesare unei decizii.
(3) Recunoaşterea formelor – prin *Rețele Neuronale Artificiale*, în care se pun în relație un set de caracteristici ale unor obiecte (reale sau abstracte) cu un set de decizii, pe baza unui model conexionist.
(4) Soluționarea problemelor de optim cu tehnici evolutive – prin *Algoritmi Genetici*, în care mulțimea soluțiilor este privită ca o populație de cromozomi în care au loc mutații şi selecții spre adaptarea optimă la un set de restricții date.

Inițial, inteligența era considerată capacitatea de a emite judecăți, iar de aici materialismul computațional a emis următoarea :secvență practică:

(A) inteligență → *raționament* → **prelucrare simbolică** → calculabilitate,

care fundamentează „Sistemele Bazate pe Cunoştințe" (în engleză Knowledge Based Systems - KBS). Totuşi, acțiunile „inteligente" ale lumii vii nu se reduc toate la raționament (improprie, după bunul simț, melcilor sau chiar pisicii spre exemplu). Ființele vii în principal reacționează la stimuli din mediu:

(B) inteligență → *reacție* → **prelucrare subsimbolică** → calculabilitate,

care fundamentează prelucrările de „Soft-computing", adică prin tehnici Fuzzy, Rețele Neuronale Artificiale şi Algoritmi Genetici.
Am putea considera prima abordare (A) ca judecata emisă de om prin „conştient" iar a doua abordare (B) ca reacția instinctivă sau intuitivă a omului prin „subconştient".

Cercetarea şi aplicațiile Inteligenței Artificiale se dezvoltă în principal către următoarele arii de interes:

- Demonstrarea teoremelor – ca modalitate de a valida un adevăr (o teoremă) prin găsirea unei secvențe deductive de alte adevăruri de la cel inițial nevalidat (al teoremei) la altul final validat anterior sau prezentat axiomatic. Utilitatea demonstrării automate a teoremelor este legată de evitarea rezolvării unor probleme dificile doar pentru contexte sau date particulare – prin algoritmi realizați ad-hoc de programatori „inteligenți" sau prin păreri exprimate de experți umani.
- Jocuri ale minții – şah sau alte jocuri în care „campioni" umani sunt provocați de maşini „inteligente" care de fapt rulează un program dedicat jocului respectiv.

- Analiză de tip Expert şi consultanță în domenii aplicative – prin care se prelucrează informații de tip simbolic şi se emulează raționamente în contexte complexe, cu cunoştințe incomplete sau variabile: diagnoză medicală şi tehnică, predicție.
- Planificarea comportării prin analiză scop-mijloc – pentru a crea şi planifica secvențe de acțiuni, privite ca mijloace, care servesc unor scopuri direct legate de mijloace, de exemplu pentru mişcarea şi comportarea roboților.
- Înțelegerea şi utilizarea limbajului natural vorbit şi scris – necesar sistemelor de traduce automată, comandă şi exprimarea vocală în interacțiunea cu maşina, înțelesului conținutului unui mesaj.
- Percepție acustică şi vizuală, recunoaşterea formelor – pentru identificare de către maşină a obiectelor şi fenomenelor pentru a le comunica omului sau pentru asistarea sa în luarea deciziilor.
- Auto-învățare şi auto-reproducere – pentru comunicarea maşină-mediu şi pentru replicarea unor obiecte (reale sau virtuale) în scopul susținerii unei utilități umane.

3.5 Rezumat

Prelucrarea datelor este funcția de bază a unui calculator şi este descrisă într-un program prin comenzile înscrise în acesta. În orice program se întâlnesc câteva categorii de bază de comenzi, ce pot fi reduse la: atribuire (obținerea unei valori din altele existente), decizia (ramificarea fluxului de comenzi urmare a unei decizii), repetiția (execuția de mai multe ori a unei secvențe de comenzi – până la îndeplinirea unei condiții de oprire). Fiecare categorie are o exprimare codificată, în scris, prin care se poate descrie textual o comandă specifică, utilă unei prelucrări anume. Aceste exprimări textuale se numesc programe (sau subprograme) şi au o structură specifică ce trebuie respectată de cel care scrie programul (programatorul). Diferența dintre un program şi un subprogram este aceea că primul poate rula independent (lansat sub sistemul de operare) iar al doilea doar apelat în cadrul altui program sau subprogram. Prelucrările din program (sau subprogram) se exprimă printr-un algoritm (indicând succesiunea de comenzi, decizii şi repetiții) acesta fiind descris sub formă grafică (organigramă) sau textuală (pseudocod). Capitolul prezintă comenzi elementare în limbaje de programare de nivel mediu, pentru că acestea dau o imagine clară a modului cum se poate descrie textual un program - ca un mod general de codificare a prelucrărilor. Se indică apoi categorii uzuale de prelucrări efectuate cu ajutorul calculatorului, de la calcule matematice până la prelucrări prin tehnici de inteligență artificială.

3.6 Teme de control

1. Descrieți modul în care expresia reprezintă o prelucrare, prin elementele ei.
2. În ce mod este expresia legată de instrucțiunea de atribuire?
3. Dați exemplu de o instrucțiune de decizie, cu elementele specifice de codificare.
4. Dați exemplu de o instrucțiune de repetiție, cu elementele sale specifice la codificare.
5. În ce situații sunt utili operatorii relaționali la scrierea unui program?
6. Prin ce se deosebesc operatorii unari de cei binari? Dați câte un exemplu (argumentat) pentru fiecare.
7. De ce este utilă folosirea subprogramelor în cadrul programelor?
8. Ce rol au numele programului şi corpul acestuia?
9. Prin ce se deosebesc instrucțiunile structurate de cele simple?
10. Indicați două metode de elaborarea a algritmilor cu speficiul lor.
11. Ce prelucrări de birou sunt cele mai des întâlnite în mod uzual şi cu ce categorii de programe se execută?

12. În ce categorie de inteligență se pot încadra prelucrările rețelelor neuronale artificiale şi prelucrările de logică vagă (fuzzy)?
13. Dați trei exemple de arii de interes în care se pot folosi aplicații de Inteligență Artificială.

Exemple de răspuns

(6) Operatorii unari se aplică asupra unui singur operand, iar cei binari la doi operanzi.
Exemple: - (minus) este operator unar care transformă UN număr pozitiv în numărul negativ simetric față de 0; + (plus) este operator binar care sumează DOUĂ numere.

(12) Prelucrările rețelelor neuronale artificiale şi cele de logică fuzzy se pot încadra în prelucrări subsimbolice sau de Soft-computing.

4 Realizarea programelor şi programe suport

Funcţionarea unui sistem de calcul presupune existenţa pe suportul fizic (*echipament*) a secvenţei de comenzi (*program*) care indică ce prelucrări suportă valorile de interes (*date*). În primele capitole s-au discutat cele trei entităţi, privind:

(I) structura constructivă şi funcţională a echipamentului de calcul;
(II) reprezentarea datelor referitoare la diverse tipuri de informaţii;
(III) comenzile elementare (instrucţiuni) şi modalităţile de soluţionare generică a problemelor (algoritmi).

Totuşi, acestea sunt doar instrumente puse la dispoziţia omului de Tehnologia Informaţiei şi Comunicaţiilor dar rezolvarea completă a unei probleme noi (necunoscute) presupune abordarea tuturor aspectelor ce apar în situaţii reale, ba chiar mai mult – elaborarea unei soluţii generice pentru toate (sau cât mai multe din) problemele similare. Sunt necesare adică:

(IV) analiza problemei (sau clasei de probleme) şi elaborarea unei soluţii generale adecvate (analiza şi soluţionarea problemei);
(V) înscrierea comenzilor de găsire a soluţiei cu prevederea tuturor situaţiilor ce pot apare, apoi furnizarea acestora calculatorului (proiectarea şi realizarea programului);
(VI) validarea soluţiei şi utilizarea ei pentru situaţii concrete din clasa de probleme .

În acest capitol se vor aborda chestiuni ce privesc acţiunile (IV), (V) şi (VI), cu metodele sistematice ce stau la baza lor, cu mijloacele prin care aceste acţiuni se pot realiza eficient (sau automat) şi cu chestiuni legate indirect de utilizarea soluţiei – cum sunt documentarea, întreţinerea sistemului, dezvoltarea sa. În principal, scopul tuturor acestor acţiuni este realizarea programului de calculator, adică *programarea*.

Programul este o secvenţă de comenzi exprimată într-un mod codificat şi care poate fi interpretat şi executat de către calculator.

După cum s-a arătat şi la §3.2.3, un program nu prezintă doar secvenţe (succesiuni) de comenzi ci şi ramificaţii sau repetiţii dar *secvenţa* este esenţială fiindcă reflectă modalitatea în care omul concepe rezolvarea unei probleme – paşi succesivi, cu o singură operaţie la un moment dat. Chiar şi procesorul este astfel realizat constructiv (indiferent cât de perfecţionat ar fi) pentru a executa o singură comandă al un moment dat, iar mai multe comenzi în serie (secvenţă). Un sistem de calcul poate executa mai multe comenzi simultan doar dacă are mai multe procesoare (este sistem multi-procesor sau maşină paralelă).

Pe de altă parte, la rezolvarea unei probleme cu ajutorul calculatorului apar situaţii complexe, colaterale problemei de bază, ce trebuie şi acestea rezolvate, cum sunt: condiţii limită ale contextului de lucru sau ale valorilor de intrare, greşeli posibile pe care operatorul uman le poate face (din necunoaştere sau neatenţie); apoi prezentarea rezultatelor este importantă: tabele, grafice, sunete, etc. Se constată deci că rezolvarea unei probleme (chiar foarte simplă) implică de fapt rezolvarea multor altor probleme pentru utilizarea de către om a soluţiei.

Aplicaţia este un set de programe reunite şi interdependente, care se prezintă într-un mod unitar şi oferă soluţii pentru o clasă de probleme date.

Sistem Informatic este un set de aplicaţii care funcţionează pe o structură de echipamente interconectate, exploatate de un număr mare de oameni către o utilitate complexă.

Ca exemple se amintesc sisteme informatice pentru: rezervarea biletelor de avion sau tren, urmărirea stocurilor de produse ale unei companii, evidenţa proceselor într-un tribunal. În cele ce urmează se prezintă chestiuni legate de programarea calculatoarelor, vizând realizarea completă şi eficientă a aplicaţiilor şi a sistemelor informatice.

4.1 Problematica programării

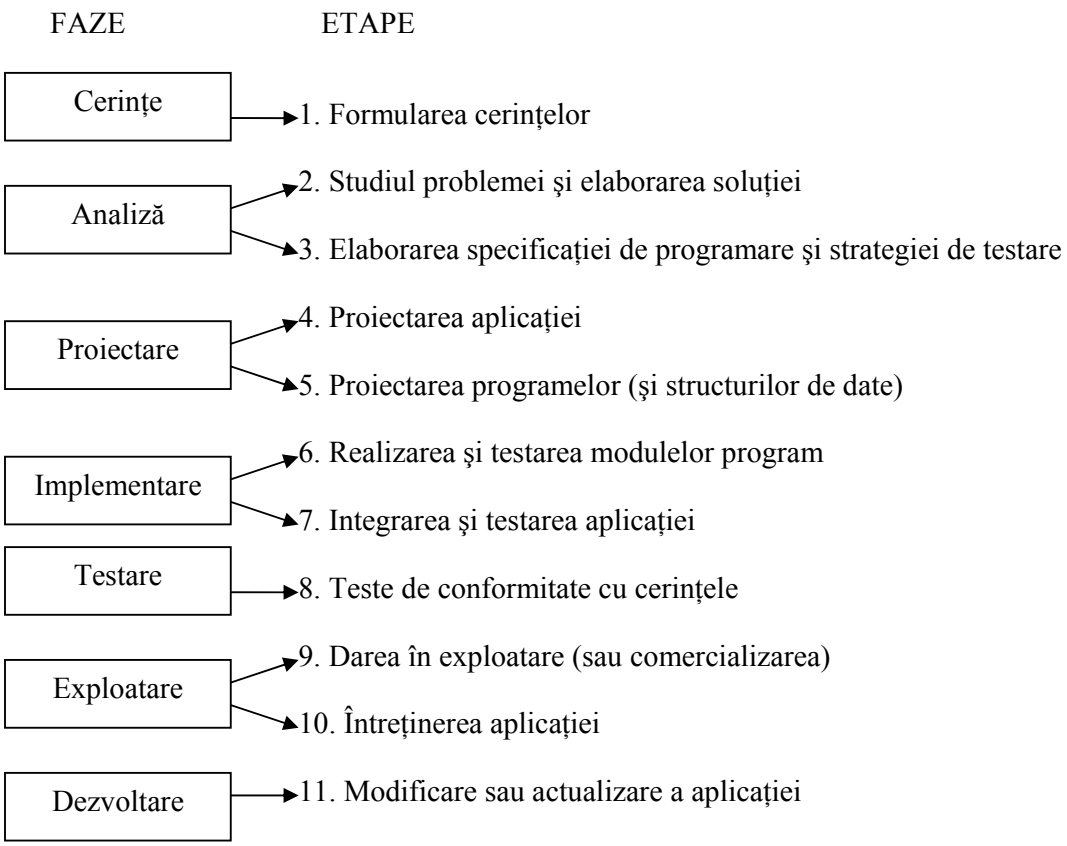
Programarea este un proces de înscriere a instrucțiunilor prin care se rezolvă o problemă şi care pot fi convertite mecanic în acțiuni ale unui sistem de calcul.

Programarea este dependentă de limbajul de codificare a instrucțiunilor, de cunoştințele programatorului privind regulile structurale (sintaxa) şi semnificația (semantica) elementelor de limbaj, de modul cum se execută (clar, comentat şi documentat) şi de modul general de gândire a soluționare a problemelor (metode de analiză şi implementare). De aceea, mult timp, programarea s-a considerat o „artă", programatorul fiind o persoană care se poate exprima pe sine (ca mod de gândire, eleganță în exprimare, eficiență a soluției) prin produsul program realizat. Fiindcă informatica a devenit o industrie, programarea este privită astăzi prin prisma eficienței muncii şi a soluțiilor, adică vizează productivitatea muncii de programare şi standardizarea utilizării produselor.

Rezultatul programării este un *produs program*. Ca orice produs, el are un ciclu de fabricație, o valoare de utilizare (şi de aici un preț), precum şi un proprietar. Spre deosebire de alte produse, un program produs de o firmă este proprietatea acesteia şi nu a cumpărătorului programului; cumpărătorul achiziționează doar dreptul de utilizare a programului şi ca atare nu îl poate modifica sau revinde.

4.1.1 Etape în ciclul de viață ale unui produs program

Programarea propriu-zisă este doar o etapă în realizarea şi existența unui program. Ciclul de viață al unui produs program cuprinde acțiuni desfăşurate de la lansarea ideii programului până la înlăturarea lui (fiind perimat sau inutil). Etapele din viața unui program sunt grupate în faze cu evoluție ciclică.



În continuare se vor descrie etapele din ciclul de viață, indicând *metodologii* sau abordări sistematice care cresc eficiența muncii în fiecare etapă. O metodologie în domeniul analizei şi proiectării unei aplicații (sau a unui sistem informatic) se referă la următoarele aspecte:

- *modele* – ca viziuni conceptuale asupra obiectelor şi activităților într-un domeniu dat,
- *metode* – ca acțiuni ce determină obiecte şi activități concrete din problema de rezolvat,
- *instrumente* – ca mijloace de lucru software care ajută analiza şi/sau proiectarea aplicației.

4.1.1.1 Formularea cerințelor

În această etapă se enunță de fapt problema ce se doreşte rezolvată cu calculatorul, şi constituie motivul pentru care este necesară aplicația şi se enunță o soluție de principiu, care rezultă din chiar în formularea cerințelor noului program sau sistem informatic. Adesea, beneficiarul aplicației nu ştie exact ce doreşte de fapt şi nici posibilitățile pe care un program i le poate oferi. De aceea, ciclul de viață se poate relua, chiar cu reformularea cerințelor – spre a fi în acord cu realitatea sau cu disponibilul financiar pentru produsul program.

La formularea cerințelor accentul cade pe problema de rezolvat, pe scopurile şi beneficiile urmărite prin aplicația vizată. Totodată, se face o analiză preliminară a resurselor financiare şi umane, a mijloacelor existente şi necesare. Nu există o metodologie specifică acestei faze, dar ea este desfăşurată, în general, ca un interviu; de aceea, abordările sistematice privesc etapele unui interviu tehnic, orientat pe scopurile şi resursele vizate, pe situația existentă şi viitoare. Urmare a formulării cerințelor, rezultă şi dacă problema se poate rezolva cu un *program*, o *aplicație* (mai multe programe integrate) sau un *sistem de informatizare* (o structură de echipamente şi programe interconectate)

Persoanele implicate în formularea cerințelor sunt beneficiarii – specialişti în domeniul pentru care se doreşte programul (care cunosc problema de rezolvat şi propun soluția folosind un sistem informatic), informaticieni (care cunosc instrumentul informatic şi propun resursele necesare rezolvării problemei) persoane de conducerea firmei beneficiare (care cunosc posibilitățile firmei şi alocă resursele necesare). Formularea cerințelor se încheie cu un „Studiu de fezabilitate” care – dacă este aprobat de beneficiar, duce la constituirea echipei de analiză în detaliu a problemei.

4.1.1.2 Analiza problemei

Această etapă începe cu „Studiu şi elaborarea soluției problemei”, prin care se evaluează situația existentă, se parcurg metode sau soluții deja aplicate în situații similare, apoi se stabilesc acelor metode (eventual şi algoritmii) care permit rezolvarea conceptuală a problemei. Studiul efectuat descrie problema (sau sistemul țintă) din patru puncte de vedere:

i) Viziunea externă (specificația) – asupra scopurilor aplicației;
ii) Viziunea organizațională (structurală) – asupra modului de realizare a aplicației;
iii) Viziunea comportamentală (temporală) – asupra evoluției dinamice a aplicației;
iv) Viziunea asupra resurselor – hardware (echipamente de prelucrare şi transfer), software (alte programe necesare aplicației), resurse umane (implicate în operarea şi utilizarea aplicației), resurse financiare (sume estimate pentru realizarea aplicației).

Pentru realizarea unei analize precise şi complete, urmată de elaborarea sistematică a soluției, se recomandă respectarea unei *metodologii de analiză*, care se este, de obicei, specifică modului de proiectare şi dezvoltare a aplicației. La baza celor mai multe metodologii stă conceptul de diagramă „Entitate-Relație”, care constă într-o reprezentare grafică, intuitivă, a obiectelor şi legăturilor dintre ele în problema reală dată. De asemenea, abordările de analiză pot evolua „de la mic la mare” (*bottom-up*, de jos în sus, de la amănunt la general) sau „de la mare la mic” (*top-down*, de sus în jos, de la general la amănunt).

Se amintesc pe scurt câteva metodologii, cu specificul lor:

a. *MERISE* – orientată spre aplicații pentru Baze de Date. Privește sistematic trei niveluri: conceptual (CE se dorește), Logic și Organizațional (CINE, CE face și UNDE), Fizic și Operațional (CUM face). Prin această metodologie se elaborează modele conceptuale pentru date și pentru prelucrări, ce vor fi apoi detaliate la faza de proiectare a aplicației.
b. *OMT* (Object Modeling Technique) – orientată pe obiecte, prin care se identifică în problema dată obiectele și metodele asociate. Ca etape parcurse se amintesc: modelarea obiectelor, modelarea dinamică (a evoluției obiectelor), modelarea funcțională (a funcțiilor structurii de obiecte).
c. *UML* (Unified Modeling Language) – orientată pe obiecte și aplicabilă în aproape orice domeniu. Obiectele utilizator și cele din structura sistemului se modelează prin: diagrame „use-cases" (scopul actorilor), diagrame „class" și „objects" (proprietăți și metode din structura ierarhică a sistemului), diagrame „sequence" și „collaboration"(mesaje și interacțiuni între obiecte), diagrame „state" și „actvity" (tranziții și roluri ale obiectelor), ce vor fi urmate în etapa de proiectare de diagrame „components" și „deployment" (implementare și configurare efectivă).

Fiecare metodologie are astăzi instrumente software adecvate, pentru asistarea experților umani în modelarea soluției și a programului sau sistemului de informatizare. Prin aceste instrumente se pot elabora sistematic structuri de obiecte conceptuale, care se reprezintă ca diagrame și scheme bloc fizice și funcționale ale viitoarei aplicații.

Persoanele implicate în această etapă sunt analiști în domeniul problemei (adică specialiști cu experiență și suficiente cunoștințe pentru a elabora o soluție viabilă și în detaliu), analiști de sistem (adică informaticieni cu experiență în tipul de probleme din care face parte cea de rezolvat), beneficiarul și utilizatori obișnuiți la sistemul existent (care dau detalii asupra situației existente și problemei de rezolvat, pretind un mod de funcționare a aplicației și un anume mod de prezentare a rezultatelor). Documentele care rezultă din această etapă sunt „Specificația de proiectare" (descrierea de principiu a informațiilor și prelucrărilor) și „Strategia de testare" (care prevede modurile în care se vor testa modulele și întreg ansamblul, precum și datele de test – cu rezultatele așteptate).

4.1.1.3 Proiectarea aplicației

Etapa de proiectare („design") se referă la structurarea efectivă a blocurilor software cu indicarea rolurilor, interacțiunilor și resurselor fiecăruia. Activitatea de proiectare implică *abstractizarea faptelor* ce au rezultat în etapa de analiză, pentru *modelarea informațiilor și acțiunilor* necesare rezolvării problemei date. Procedura de abstractizare elimină faptele irelevante și le accentuează pe cele esențiale, iar procedura de modelare reprezintă informații și acțiuni într-un mod specific. Pentru un produs informatic, modelul poate fi *formal* (adică exprimat prin simboluri, de exemplu prin formule) sau *procedural* (adică exprimat prin cuvinte ca o rețetă de bucătărie).

Metodologia aplicată la etapa de proiectare este puternic dependentă de modalitatea de programare. De aceea, etapele de proiectare și implementare sunt strâns legate, uneori chiar suprapuse iar această legătură provine din modul cum este gândită, chiar de la etapa de proiectare, realizarea efectivă (implementarea) aplicației pe întregul ei și pe fiecare program în parte. Între metodologii se amintesc două mai importante: proiectarea obiectuală (pentru aplicații în care se pot discrimina obiecte din lumea reală ce sunt manipulate de aplicație) – cu utilizare mai frecventă în domenii tehnice și care simulează realitatea, proiectare cu baze de date (pentru aplicații de gestiune a resurselor de orice fel) – cu utilizate frecventă în economie și administrație.

Pe lângă partea software, la această etapă se proiectează și *structura de echipament*, privind: structura de calculatoare și configurația fiecăruia, structura de comunicație (rețea locală, echipamente de rețea, conectarea la Internet), structura de periferice partajate (adică folosite

în comun) de mai mulți utilizatori (imprimante sau mese de desen scumpe, interfețe de proces pentru culegerea datelor sau comanda din / către instalații). Se proiectează tipul şi configurația *sistemelor de operare* - strâns legat de structura de echipamente şi de scopurile aplicației.

Persoanele implicate în această etapă sunt: analişti de sistem (informaticieni cu pregătire specială în folosirea unui instrument de proiectare şi implementare programelor), ingineri hardware şi ingineri de sistem (care proiectează structura de echipamente şi programe), conducători de proiect (specialişti în domeniul țintă sau în informatică, care cunosc modul de organizare a activităților complexe precum şi domeniul țintă). Documentele elaborate la finalul etapei sunt „Specificația de programare" (indică structura de module şi acțiunile apoi datele necesare fiecărui program), „Planificarea lucrărilor de implementare", „Inventarul resurselor necesare" (financiare, umane şi materiale) pentru realizarea noului program sau sistem de informatizare,

4.1.1.4 Implementarea şi testarea aplicației

Activitatea esențială a acestei etape este programarea. Se vorbeşte adesea de „programarea calculatoarelor" subînțelegând toate activitățile implicate de aceasta, poate fiindcă programarea este activitatea prin care efectiv echipamentul de calcul devine din funcțional (fără programe este „fier mort"). În sine, *programarea* constă în codificarea operațiunilor pe care calculatorul trebuie să le execute către atingerea unui scop dat (calcul matematic, retuşarea şi afişarea unei imagini, sau mişcarea brațului unui robot). După cum se constată, programarea este doar partea de realizare efectivă a programului, care însă necesită multe alte activități anterioare şi posterioare. Fazele realizării unui program (v.) sunt:

(I) Înscrierea *programului sursă* – prin care se descriu acțiuni (folosind un limbaj de programare) într-un text scris cu un editor de texte. Atât limbajul cât şi modul de realizare a programului sursă sunt apropiate obişnuințelor umane (cum spre exemplu, o rețetă de bucătărie este înscrisă ca text, într-o formă simplificată uneori chiar codificată).

(II) *Compilarea* – prin care textul sursă este „tradus" din limbajul de programare (exprimat prin cuvinte – v. §4.1.2) în limbajul maşinii (exprimat prin coduri binare – v. de ex. §2.3.3.1). Traducerea este realizată de un program special pentru limbajul de programare ales – numit *compilator*, iar rezultatul este *codul obiect* al programului.

(III) *Editarea legăturilor* – prin care în codul obiect se inserează subprograme, preluate din biblioteci de subprograme, ce descriu prelucrări uzuale, pe care programatorul le foloseşte fără a mai scrie cod (fără a scrie programul ci doar a-l apela din bibliotecă). Astfel, prelucrări care au fost doar amintite în programul sursă se înscriu efectiv în codul obiect. Rezultatul fazei este *codul executabil* al programului, adică forma binară ce poate fi încărcată direct în memoria de lucru şi poate executa operațiunile programate.

Cuvintele ce exprimă comenzi se combină în limbajul de programare respectând o sintaxă strictă (ca reguli gramaticale); programatorul poate greşi (din neatenție, din necunoşatere), astfel că textul sursă fiind greşit să nu poată fi interpretat de calculator. În acest caz, este necesară:

(IV) *depanarea programului* – care constă în modificarea textului sursă spre a fi eliminate erorile. Identificarea erorilor şi apoi verificarea programului se realizează cu ajutorul unui *depanator* (program de asistare a programatorului în activitatea de verificare a corectitudinii programului). Corectarea efectivă a erorilor constă în înscrierea corectă a cuvintelor sa a combinațiilor de cuvinte în textul sursă.

Depanatorul localizează erorile din program şi face chiar sugestii de corectură, însă aceste erori sunt legate doar de „modul de exprimare" în limbajul dat, nu de modul cum a fost rezolvată problema (soluția corectă sau nu); eliminarea erorilor de soluționare a problemei se

poate face doar prin executarea de teste pe date şi în situaţii reale, urmată de compararea rezultatelor cu cele aşteptate şi apoi modificarea algoritmilor de prelucrare.

În general, editarea, compilarea şi depanarea programului se realizează folosind un mediu integrat (un program complex cu toate aceste instrumente), spre a spori eficienţa muncii de programare. Astfel de instrumente sunt „mediile de programare" (v. §4.2.3.2) sau „instrumentele de inginerie software" CASE (v. §4.2.3.4). Scrierea efectivă a programului se numeşte codificare. Această operaţiune complexă nu se realizează doar înscriind textul în limbajul de programare ales ci se includ date, obiecte sau prelucrări „prefabricate" din biblioteci ale mediilor de programare utilizate pentru scrierea aplicaţiei. În variante mai evoluate (utilizând instrumente CASE sau instrumente RAD), se pot realiza programe prin „plasarea" unor obiecte virtuale ce reprezintă date şi acţiuni necesare rezolvării diferitor aspecte ale problemei (introducere de date, prelucrare, afişare).

Aplicaţia se implementează modular – fiecare subprogram rezultat la proiectare (şi înscris în „Specificaţia de programare") este codificat şi testat separat. La realizarea programelor se respectă principii de inginerie a programării, în scopul depanării facile şi apoi a dezvoltării coerente a fiecărui modul şi aplicaţiei. După ce modulele sunt verificate, se face *integrarea aplicaţiei*, adică se instalează toate piesele software şi hardware ale aplicaţiei. Se face *testarea* ansamblul în condiţii de laborator şi se emit documentele de conformitate cu cerinţele (dacă sunt respectate sau nu, care din cerinţe nu au fost satisfăcute şi de ce). În situaţia în care funcţiile aplicaţiei sau utilizarea acesteia nu sunt conforme cerinţelor, se decide dacă, şi pentru care din cerinţe, se reiau fazele de analiză, proiectare şi apoi cele de implementare cu testare.

Similar fazei de proiectare, pentru structura de echipamente se procedează la achiziţionarea, instalarea şi testarea fiecărui echipament şi a întregului sistem, format din calculatoare, reţea şi echipamente de interconectare, periferice în reţea, alimentare cu energie electrică, spaţii de securizare a echipamentelor sensibile şi stocare a suporturilor cu date.

Persoanele implicate în aceste faze sunt: analişti programatori (elaborează structurile conceptuale de module sau obiecte şi stabilesc datele şi prelucrările pentru fiecare din ele), programatori (realizează codificarea programelor), ingineri electronişti, electricieni, alţi tehnicieni (pentru instalarea echipamentelor şi, eventual, amenajarea spaţiilor), ingineri de sistem (pentru instalarea sistemelor de operare şi integrarea aplicaţiilor), precum şi beneficiari sau utilizatori (pentru testarea utilizării aplicaţiilor şi certificarea conformităţii cu cerinţele). Documentele elaborate în finalul acestei etape sunt: „Programe sursă" ale aplicaţiei şi fişierelor de comandă, „Documentaţia aplicaţiei" (care descrie structura de module, funcţionarea şi configurarea aplicaţiei), „Manualul de utilizare", „Fişe de testare" (care constată conformitatea utilizării cu cerinţele). Darea în exploatare a aplicaţiei se face după o testare la utilizator (de obicei de 72 de ore), şi dacă aceasta a decurs cu succes se încheie un „Proces verbal de recepţie". Acest document încheie ciclul de realizare al aplicaţiei; orice alte modificări solicitate şi realizate după acest moment se consideră lucrări separate, pentru care se parcurg din nou etapele (de la analiză până la implementare şi testare).

4.1.1.5 Exploatarea şi dezvoltarea aplicaţiei

După ce aplicaţia a fost testată şi recepţionată de către beneficiar ea intră în exploatare, adică este utilizată pentru scopul pentru care a fost realizată. „Darea în exploatare" este faza în care personalul utilizator urmează cursuri de pregătire pentru folosirea aplicaţiei şi, eventual, conducerea asigură cadrul organizatoric (personal specializat, spaţii şi regulamente de lucru) pentru aplicaţia sau sistemul în cauză.

Pe durata exploatării aplicaţiei (sau a sistemului de informatizare) pot apare diverse probleme, care trebuie rezolvate spre a se asigura o funcţionare corectă. Între probleme se pot aminti: actualizarea unor date de tip parametric ale aplicaţiei (de exemplu la modificarea legislaţiei

legată de o aplicație de contabilitate), configurare periodică, administrarea resurselor sistemului (imprimante, discuri, conturi utilizator), rezolvarea unor incidente de operare sau ce apar după un accident (defect) sau după modificări în echipamente. Acțiunile de rezolvare a unor asemenea probleme se pot reuni în activitatea de *întreținere a aplicației*. Persoanele implicate în această activitate sunt: ingineri de sistem (asigură configurarea corectă a sistemului de operare şi a programelor de aplicație), administratori de baze de date (asigură configurarea şi asistă utilizatorii în utilizarea corectă a bazelor de date) administratori de rețea (asigură configurarea şi menținerea bunei funcționări a echipamentelor şi programelor de comunicație), ingineri şi/sau tehnicieni de întreținere echipament, utilizatori obişnuiți şi utilizatori „privilegiați" – ultimii având de fapt sarcini speciale, de exemplu gestiunea resurselor grupului de lucru, servicii de configurare specifică grupului de lucru; calificativul de „privilegiat" se referă la drepturile (şi răspunderile) extinse pe care le au privind accesul la date şi programe.

Exploatarea aplicației – în forma în care a fost achiziționată, are loc până la apariția unei versiuni mai perfecționate (adică o dezvoltare - în engleză „up-grade") sau până la inutilitatea ei (datorită apariției pe piață a unor noi produse sau prin dispariția scopului aplicației). În măsura în care prin modificarea aplicației se pot obține caracteristici mai performante, se poate intra într-o etapă de *dezvoltare a aplicației*, în care se repetă toate etapele – de la analiză la implementare, parcurse la realizarea aplicației. Evident, cea mai marea parte a programelor din aplicația curentă nu ar trebui să sufere modificări ci doar cele care nu mai sunt de actualitate sau necesită perfecționări.

4.1.2 Limbaje de programare

În capitolele 2 şi 3 s-au amintit unele limbaje de programare în contextul declarării tipurilor de date iar apoi a tipurilor de comenzi necesare descrierii prelucrărilor prin sisteme de calcul. De fapt, acestea sunt cele două aspecte generice prin care omul comunică maşinii ce are de făcut: cu ce (datele) şi cum (comenzile). La primele calculatoarele din primele generații (din 1948 şi până în 1980) se puteau introduce datele şi comenzile direct în cod binar, prin comutatoare. Acest mod de lucru era extrem de greoi şi în plus era accesibil doar celor care cunoşteau bine maşina şi reprezentările interne ale datelor şi comenzilor (iar varietatea maşinilor de calcul devenise deja foarte mare). Un limbaj de programare apropiat omului oferă independență programatorilor față de tipul maşinii de calcul şi permite acestuia să se concentreze asupra rezolvării problemei, nu asupra maşinii.

Un ***limbaj de programare*** este un set de cuvinte cu semnificații precise, care se pot combina după reguli stricte pentru a exprima comenzi şi a descrie date necesare unui tip de prelucrare.

4.1.2.1 Limbaje de programare uzuale

Deja în capitolele 2 şi 3 s-au amintit limbajele de programare C, Pascal şi Java pentru a exemplifica cele două aspecte importante: descrierea datelor şi structurilor de date, exprimarea comenzilor elementare (instrucțiuni). Se prezintă mai jos, în fiecare paragraf, câte un limbaj de programare, ordonate după gradul de extindere şi frecvența de utilizare de către programatori.

1. **C** (pronunțat ca în engleză „si") este dezvoltat de Kernigan şi Ritchie la Bell Laboratories în anul 1972, fiind ulterior limbajului denumit B. C a fost dezvoltat pentru crearea de programe apropiate de maşină (sisteme de operare, driver-e) fiind legat de sistemul de operare UNIX. Popularitatea sa cât şi standardizarea de către ANSI l-au impus ca un limbajul cel mai larg acceptat de programatori. Varianta **C++** este un limbaj de programare orientat pe obiecte (v. §4.2.1), fiind extins cu directive pentru crearea şi manipularea obiectelor. Există alte diverse variante îmbunătățite, ca

Visual C++ (cu mecanisme de creare a interfețelor grafice şi lucrul în rețea), **C#** (pronunțat „si şarp", cu servicii pe Internet în categoria .net – „dot net").

2. **Java** (pronunțat „giava") este dezvoltat de firma SUN Microsystems în scopul declarat de a realiza aplicații pentru Internet prin compilator şi biblioteci gratuite, fiind similar limbajului C++ (orientat pe obiecte şi instrucțiuni identice) însă compilarea produce un „cod de octeți" (nu cod executabil). Este extins pentru lucrul cu fire de execuție (secțiuni de program ce pot rula independent), tratarea excepțiilor (erori, sau întreruperi), securitate (execuția se face prin intermediul „Java Virtual Machine" care interpretează şi controlează „codul de octeți" spre a nu permite acțiuni ostile – de ex. prin acces direct la resursele sistemului), portabilitate (poate rula pe orice calculator care prezintă „Java Virtual Machine").
3. **JavaScript** este un limbaj scriptural prin care se adaugă paginilor web facilități de animație şi de interacțiune cu utilizatorul (diferit şi folosit mult mai des decât Java pentru programarea părții client web). Este standardizat sub numele de ECMAscript.
4. **HTML** (Hyper Text Markup Language) este un limbaj scritpural de descriere a documentelor, folosind marcaje ce specifică acțiuni de formatare înscrise chiar în textul (conținutul) documentului. Paginile Web sunt descrise prin acest limbaj care permite în plus legături cu alte pagini distribuite spațial pe alte maşini (site-uri); se poate astfel considera că textul documentului nu mai este înscris pe o foaie cu două dimensiuni ci prezintă şi adâncime – către alte texte (este hiper-text). Informațiile din pagina cu text şi imagini sunt descărcate („download") şi afişate pe maşina utilizatorului (v. §6.2.1.1). În aceste pagini, acțiunile de formatare privesc modul de scriere a textului (litere îngroşate, cursive, etc.), structura documentului (denumiri de capitol, paragrafe, tabele) şi interacțiunea cu utilizatorul (de exemplu prin formulare ce permit transferul de date către site – „upload").
5. **BASIC** este un limbaj creat inițial pentru începători (chiar dacă de fapt era destul de greoi şi nestructurat). Ulterior, limbajul a fost perfecționat iar în versiunea **Visual Basic** este orientat pe obiecte şi prezintă instrumente de programare vizuală a interfețelor utilizator grafice (v. §4.4), fiind folosit şi pentru aplicații în Internet sau lucrul cu baze de date.
6. **XML** (eXtensible Markup Language) este o extensie a limbajelor din familia SGML, în care marcajele nu mai sunt predefinite şi stricte ci pot fi definite chiar de programator pentru a descrie diverse tipuri de date dar şi prelucrări. Este utilizat în principal pentru partajarea informațiilor şi textelor structurate în Internet.
7. **SQL** (Structured Query Language) este creat de firma IBM şi adoptat ca standard de ISO (International Standards Organization). SQL (pronunțat „sicuăl") este un limbaj de programare declarativ pentru utilizare în baze de date relaționale, oferind exprimări simple şi intuitive de interogare precum şi manipularea tabelelor, formularelor şi rapoartelor în aplicații cu baze de date.
8. **FORTRAN** (FORmula TRANslation) este primul limbaj de nivel înalt, creat de John Backus pentru a asigura independența de maşină, utilizat mai ales pentru calcule ştiințifice. Ca şi alte limbaje de programare a evoluat, având şi facilități visuale.
9. **COBOL** (COmmon Business-Oriented Language) dezvoltat în anii '60 şi larg folosit pentru aplicații de gestiune economică şi administrativă. Este un limbaj procedural, compilat, care în primele versiuni era un bun exemplu de limbaj nestructurat; ulterior a fost perfecționat dar pe parcurs este eliminat de către mediile de baze de date – instrumente simple şi puternice pentru aplicații de gestiune.

10. **PHP** este un limbaj scriptural similar cu PERL, ce poate fi înglobat în codul HTML al paginilor Web. Este folosit pentru aplicațiile pe partea server web, pentru construirea paginilor web dintr-o bază de date SQL (Oracle, MySQL). Prezintă una din cele mai mari biblioteci „open-source" (cod public şi liber a fi utilizat, modificat sau distribuit).
11. **LISP** (LISt Processing) este primul limbaj funcțional, fiind dezvoltat de John McCarthy de la MIT în anii '50. de orientat pe structuri de date (liste). Este folosit pentru aplicații de Inteligență Artificială, în probleme ce implică raționament calitativ sau multe date eterogene – potențial eronate dar care trebuie refăcute, probleme de planificare şi antrenare (învățare) a maşinii.
12. **Prolog** este un limbaj care exprimă premize (fapte de start într-un raționament – denumită „logica"), prin care interpretorul Prolog (denumit „control") încearcă să demonstreze o porpoziție declarată peste premize
13. **Asamblare** este limbajul maşinii într-o notație inteligibilă omului. La compilarea unui program scris în limbaj de asamblare, conversia se face direct în coduri binare, fiindcă instrucțiunile limbajului se referă la acțiuni elementare ale procesorului (încărcare de regiştri, salturi condiționate, operații pe bit). Este utilizat pentru scrierea programelor de control direct al perifericelor sau pentru operații legate de echipament.
14. **Bourne Shell** este un limbaj scriptural care permite crearea de fişiere de comenzi (în loturi - „batch") pentru sistemul de operare UNIX (alte limbaje similare sunt *sh*, *bash*, *ksh*, *csh*). În general, orice sistem de operare prezintă limbajele de comandă (limbaje shell), necesare lucrului imediat cu calculatorul, comenzile fiind executate de un „interpretor de comenzi" furnizat cu sistemul de operare – v. § 4.3.1.1.

Se face observația că limbajul de „Asamblare", înscris în lista de mai sus, este o categorie de limbaje nu un limbaj anume, însă datorită extinderii calculatoarelor de tip IBM PC se subînțelege, în general, ca limbajul familiei de procesoare Intel 80x86; există limbaje de asamblare pentru diferite procesoare (de la diferiți producători şi pentru diferite generații): Intel 80x86, Motorola 680x0, PowerPC, etc.

Utilizarea unui limbaj sau altui limbaj de programare depinde de scopul şi tipul programării (indicate succint în inventarul de mai sus) dar şi de obişnuința sau preferințele programatorului.

4.1.2.2 Clasificări ale limbajelor de programare

O clasificare a limbajelor de programare des întâlnită, consideră *nivelul limbajului*, relativ la apropierea (sau depărtarea) formulărilor de limbaj față de limbajul maşinii (mai precis al procesorului, ca dispozitiv electronic, binar).

În tabelul de mai jos se prezintă succint clasificarea după nivel a limbajelor, după gradul de apropiere de nivelul uman, deci mai simplu de utilizat de către om.

Nivelul limbajului	Caracteristici ale limbajului	Exemple uzuale
Jos	Instrucțiunile sunt apropiate de limbajul maşinii, fiind translatate direct în instrucțiuni maşină de către asamblor (un compilator simplu)	Asamblare
Mediu	Instrucțiunile sunt transpuse în limbaj de asamblare prin compilator, oferind în plus biblioteci şi servicii de configurare a resurselor maşinii la execuție	C, BASIC, Pascal, COBOL
înalt – compilat	Instrucțiunile sunt transpuse într-un cod intermediar folosind un interpretor, permițând astfel controlul codului şi portabilitatea sa pe orice maşină	Java, Visual Basic, Visual C

înalt - scriptural	Instrucțiunile pe linii de program sunt interpretate şi executate fiecare în parte; liniile se pot grupa în loturi şi executate ca „fişier (text) de comandă”	Bourne Shell, HTML, Perl,
foarte înalt	Structura limbajului este apropiată limbajul uman sau descrie o metodă de implementare; sunt limbaje declarative sau pentru Inteligența Artificială	SQL

O altă clasificare se poate face după *modul de declarare a tipurilor de date*, în limbaje cu: tipuri statice de date – tipurile declarate sunt stricte şi verificate de compilator (exemple C, C++, Java, Pascal), tipuri dinamice de date – în care datele de tipuri diferite pot fi interschimbate, interpretorul nesemnalând eroare la o dată nouă cu tip nedeclarat (exemple Lisp, JavaScript, Prolog).

4.1.2.3 Compilatoare şi interpretoare

Limbajele de programare permit realizarea codului ce va fi executat după traducerea sa în limbajul maşină prin:

- *Compilare* –traducerea are loc pentru întregul set de comenzi (descrise ca un tot unitar, într-un „program”). Limbaje din această categorie sunt C, C++, Java, Pascal, BASIC, FORTRAN, COBOL.
- *Interpretare* – traducerea are loc linie cu linie (câte o comandă la un moment dat), de la prima până la ultima din setul dat. Limbaje din această categorie se numesc limbaje scripturale, iar textul cu comenzile se numeşte script, „fişier de comenzi” sau „lot de comenzi” (în engleză „batch”). Exemple de limbaje script sunt HTML, Perl, PHP, limbaje de comandă ale sistemului de operare („shell” – Bourne Shell, bash, csh).

Există limbaje care pot fi executate în ambele moduri dar, de obicei, au specific doar unul din ele. Limbajele script pot coexista pe aceeaşi maşină; astfel, pentru selecția tipului de limbaj efectiv utilizat, prima linie din lotul de comenzi script conține o directivă a sistemului de operare ce indică în clar limbajul la care se referă fişierul de comenzi. Avantajul compilatoarelor este acela că programul în cod executabil poate fi rulat direct pe maşină, imediat după încărcarea lui în memorie de pe suportul extern (disc). Un caz special îl constituie limbajul Java, care este interpretat după ce este „compilat” în aşa-numitul „cod de octeți” care este apoi interpretat pe orice maşină ce prezintă maşina virtuală Java („Java Virtual Machine”).

4.1.3 Ingineria programării

Termenul „inginerie” din denumirea acestui subcapitol duce cu gândul imediat la tehnică şi industrie. Aşa cum termenul „tehnologie” din acronimul TIC nu se referă la tehnologii în industria metalurgică ci doar la suma de tehnici şi mijloace din informatică (v. definiția din §1), termenul „inginerie a programării” se referă la tehnică drept abordare sistematică, de producție eficientă a programelor pe calculator (în mod industrial), nu la un domeniu ingineresc (metalurgie spre exemplu). De fapt, etapele de realizare a programelor (prezentate la §4.1.1) constituie esența acestei abordări sistematice.

Ca şi la alte produse, calitatea produselor program (a software-ului) nu este doar un deziderat de piață ci şi un scop impus de toleranțele în care produsul trebuie să-şi realizeze utilitățile. Astfel, în timp ce pentru o maşină se spălat automată toleranța de 2% în alimentarea cu detergent este admisă (şi considerată foarte bună), un sistem de contabilitate care are erori de 2% este inacceptabil. De aceea, sunt necesare şi pentru software modalități de măsurare (metrici) de evaluarea a performanțelor şi de aici a calității produselor program.

Ingineria programării este abordarea sistematică, disciplinată şi cuantificată a dezvoltării, operării şi întreţinerii produselor program.

Dezideratul ingineriei programării este satisfacerea cerinţelor clientului produsului software, respectând restricţii de calitate, cost şi durată, prin:

i) utilizarea unei metodologii clare începând cu faza de analiză, apoi cea de proiectare şi de implementare a programelor;
ii) conducerea activităţilor, desfăşurate pe parcursul proiectul, conform unor principii de management stricte („project management”);
iii) analiza şi evaluarea performanţelor codului, a structurii de module, funcţionării şi utilizării produsului;
iv) documentarea clară şi completă a funcţionării şi utilizării produsului software;
v) urmărirea produsului livrat la beneficiari şi pe întreaga durată a ciclului său de viaţă, pentru actualizarea cu versiunile noi şi îmbunătăţite.

Factori de performanţă ale produselor software sunt: funcţionalitatea (în ce măsură produsul îndeplineşte funcţiile propuse), uşurinţa în utilizare (simplitatea de învăţare şi operare), fiabilitatea (funcţionarea corectă şi robustă – tolerantă la operare sau date greşite), eficienţa (privind resurse utilizate – memorie puţină, viteză de execuţie), flexibilitate (uşurinţa de adaptare şi modificare după cerinţe), portabilitatea (posibilitatea de transfer pe alte maşini şi sisteme de operare, respectarea standardelor), întreţinerea comodă (acces la cod sursă şi compilare, modificare uşoară)

Pentru aplicaţii mari, organizarea proiectului implică un *şef de proiect* (contribuie la proiectare şi dezvoltare în proporţie de 30%, distribuie sarcini şi coordonează coechipierii), *adjunct* (planifică şi coordonează programarea şi testele, asigură calitatea produsului), *secretar de proiect* (execută sarcini administrative privind protocoale cu beneficiarul, biblioteci, gestiunea termenelor şi costurilor), *programatori* / *dezvoltatori* (specialişti în medii şi limbaje de programare sau în instrumente de dezvoltare software).

Analiza, proiectarea şi implementarea respectă o metodologie specifică (uzual din categoriile „structurată” sau „obiectuală”), iar testarea se execută atât în condiţii de laborator cât şi pe cazuri reale. Întreţinerea aplicaţiei este foarte importantă, aşa cum reiese din proporţia uzuală a costurilor pentru software: Analiza – 10%, Proiectarea – 10%, Implementarea – 10%, Testarea – 20%, Întreţinerea – 50%.

O importanţă deosebită o are documentaţia, care trebuie întocmită pe parcursul proiectului la fiecare etapă. Documentaţia se adresează dezvoltatorilor, utilizatorilor şi personalului de întreţinere. Documentaţia ce trebuie întocmită a fost deja amintită la fazele ciclului de viaţă ale programului. De remarcat că programele trebui amplu documentate (comentate) chiar în codul sursă, unde algoritmul trebuie explicat.

4.2 Tehnici şi instrumente de realizare a programelor

Istoric, în evoluţia proiectării aplicaţiilor şi a programării, au existat următoarele abordări – din care au rezultat metodologii specifice:

i) *Proiectarea* / *programarea nestructurată* – în care aplicaţia nu este gândită şi realizată modular ci ca un tot, cu salturi interne greu de controlat şi urmărit. Dezvoltarea şi întreţinerea programelor este extrem de ineficientă şi greoaie. Această abordare se poate compara cu generaţia televizoarelor cu tuburi electronice („lămpi”) şi componente interconectate prin sârme (formând uneori „ghemuri”).
ii) *Proiectarea / programarea structurată* – în care aplicaţia este divizată în module specializate pentru anumite operaţii, asamblate apoi (mai precis apelate) de către

aplicația „principală”. Abordarea se poate compara cu generația televizoarelor realizate pe module specializate (alimentare, selectare canale, sunet, etc.) montate pe placa de bază. Avantajele provin din faptul că pentru depanarea sau modificarea unui modul se lucrează numai cu acesta, nu cu întreg aparatul (respectiv întreaga aplicație). În programare se elimină salturile necondiționate, fiindcă chiar din etapa de proiectare prelucrările sunt ierarhizate astfel ca un modul să „apeleze” un alt modul specializat pentru o acțiune de amănunt anume.

La aceste abordări datele sunt analizate separat de prelucrări. În cazul ii) modulele sunt realizate prin subprograme (funcții) iar datele sunt declarate separat de prelucrări. Proiectarea se bazează pe o parcurgere *„top-down”* (de la mare la mic) a problemei, pornind de la ansamblu şi apoi trecând la părți, module; pentru fiecare din acestea se discriminează datele şi prelucrările corespunzătoare.

iii) *Proiectarea / programarea orientată obiect* – în care aplicația se construieşte din obiecte care încapsulează proprietăți şi metode – adică informații (date) şi prelucrări (operații asupra datelor). În acest mod, la analiză şi proiectare se concep clase de obiecte similare celor din lumea reală a problemei de rezolvat, cu „modul lor de utilizare”, iar rezolvarea problemei se face prin manipularea obiectelor create la implementare. Abordarea se poate compara cu viziunea utilizatorului de televizoare, în care clasa de obiecte „televizor” trebuie să dețină ecran şi legătură prin cablu (ca informații - date) şi butoane de acționare pornit/oprit, reglaj volum, comutare canale, etc.

iv) *Proiectarea / programarea cu componente* – în care aplicația se construieşte prin componente gata fabricate. Abordarea este similară construirii televizorului din circuite integrate specializate, care doar se asamblează în modul dorit spre a produce un televizor stereo sau mono, cu teletext sau fără. Aşa cum componentele fizice vin de la fabrică componentele software vin de la producători software (de ex. firma Microsoft) şi sunt utilizate de programatori pentru a crea produsul dorit.

La ultimele două abordări, proiectarea decurge *„bottom-up”* (de la simplu la complex) adică se face inventarul de obiecte sau componente la dispoziție şi apoi se „construiesc” aplicații prin asamblarea acestora. Continuând similitudinea cu producția de televizoare, generația iii) reprezintă crearea de televizoare artizanale – fiecare producător realizează televizoare în tehnologie proprie, pe când cu iv) producătorii folosesc componente standard pentru părți de televizor, oferind înfățişare şi performanțe speciale produselor proprii față de ale altor producători. Ultimele două generații permit şi stimulează *industria software*, fiindcă proiectantul şi programatorul nu mai sunt implicați în atâtea amănunte de lucru la fiecare program în parte (amănunte pe care nu le pot stăpâni perfect şi nici nu au productivitate dacă sunt multe sau necesită mulți coechipieri) ci se pot orienta pe producție, pe nevoile clienților şi pe cererea pieței. Proiectantul şi programatorul au „în spate” o industrie de componente pe care trebuie doar să ştie să le asambleze pentru a face un produs la comandă.

v) *Proiectarea / programarea cu agenți* – în care aplicația (ca agent) evoluează de sine stătător şi comunică cu alte aplicații (alți agenți) pentru a-şi îndeplini misiunea. Spre exemplu, există programe agent de căutare care pot „călători” prin Internet, se pot stabili la anumite site-uri şi comunica cu alți agenți spre a găsi informațiile pentru care au fost creați şi lansați în spațiul cibernetic.

În timp ce pentru primele patru generații aplicația se lansa şi se executa prin operarea directă a omului la program – adică inițiativa aparține omului, generația v) introduce *programe cu inițiativă*, adică programe care după lansare au existență şi acțiuni independente de omul în contul căruia execută prelucrările. Programul-agent se poate multiplica, comunică cu alți agenți şi cu „baza”, formează grupuri şi chiar stabileşte relații sociale şi limbaje de

comunicare între agenți; se poate spune că din „obiecte” informatice aceste tipuri de date devin „ființe” informatice.

La inventarul abordărilor prezentate mai sus pentru proiectarea aplicațiilor, este necesar să se adauge încă două – care nu aduc metodologii conceptual noi ci doar specifice unor instrumente frecvent folosite în realizarea aplicațiilor:

vi) *Proiectarea aplicațiilor cu Baze de Date* – în care datele se structurează în tabele, adică mulțimi de articole (v. §2.3.5.3). De fapt, un articol (ca linie din tabel) reprezintă un obiect iar tabelul colecția de obiecte de același fel din problema de rezolvat – adică tabelul reprezintă o entitate (o categorie conceptuală de obiecte). Proiectarea datelor şi prelucrărilor se realizează separat – datele ca tabele iar prelucrările ca operații cu acestea, vizând direct instrumentele software cu care se vor realiza aplicațiile.

vii) *Proiectarea aplicațiilor Web (servicii Internet)* – în care datele sunt, de obicei, pagini cu informații ce trebuie vizionate de utilizatori prin Internet sau datele provin de la utilizatori prin formulare completate de către aceştia, iar prelucrările sunt operații de navigare, afişare şi actualizare a datelor (stocate adesea în baze de date). Proiectarea aplicațiilor Web se bazează pe arhitectura Client-Server (v. §4.2.2.3), pe principii de marketing, impact estetic şi emoțional, utilizând instrumente de proiectare şi editare de pagini web (cu imagini, formulare, animații şi hiperlegături).

4.2.1 Caracteristici ale programării orientate obiect

Fiindcă cea mai mare parte a aplicațiilor actuale folosesc conceptul de obiect şi sunt realizate obiectual, se vor prezenta în continuare caracteristici ale programării obiectuale, pentru familiarizarea cu termenii şi abordările întâlnite în subcapitolele următoare.

Programarea orientată obiect (POO) se bazează pe *clase*, ca abstractizări ce reunesc datele şi prelucrările posibile asupra lor (v. §2.3.6). Un obiect realizat (instanțiat) într-o clasă dată, prezintă anume valori pentru date (identificate ca proprietăți ale obiectului) şi anume comportare (identificată prin metode de modificare a proprietăților). POO vizează, în principal, următoarele aspecte:

- Crearea şi manipularea de obiecte – prin care se modularizează acțiunile programului încă din faza de analiză, atunci când se identifică obiectele în problema reală;
- Refolosirea codului – prin care obiecte odată codificate se pot reutiliza ori de câte ori este necesar, fiind grupate în colecții (denumite biblioteci sau pachete).

Aceste deziderate se obțin ca urmare a caracteristicilor programării obiectuale, între care mai importante ale sunt:

a. *Abstractizarea* – prin care un obiect devine modelul unui „actor” ce prezintă o stare (şi o poate modifica), execută acțiuni sau comunică cu alte obiecte din sistem.
b. *Încapsularea* – prin care accesul la proprietățile obiectului se poate face numai prin metodele definite. Obiectul prezintă o *interfață* către alte obiecte, prin care se specifică modalitățile sale de manipulare.
c. *Moştenirea* – prin care o clasă de obiecte poate fi baza altor clase (denumite clase derivate), proprietăți şi metode esențiale ale primei fiind preluate în întregime de celelalte. Se realizează astfel *specializarea* claselor (şi obiectelor).
d. *Polimorfismul* – prin care o metodă a unui obiect din clasă derivată produce o comportare diferită față de cea a clasei de bază.

4.2.2 Tipuri de aplicații

Structura unui program a fost prezentată la §3.2.3.2, indicând părțile specifice ale programului principal, pentru un limbaj de programare comun (procedural şi cu tipuri statice de date) cum

sunt C, Pascal sau Java. Modul cum se construiesc aplicații complexe din mai multe programe, constituie structurile generale ale aplicațiilor, așa cum se prezintă mai jos.

4.2.2.1 Aplicații

În sine, o *aplicație* cuprinde un program principal care are rol de „dispecer" către prelucrările efective ale aplicației. Lansarea aplicației se realizează la inițiativa utilizatorului, care înscrie o comandă (în forma text) sau accesează o pictogramă într-o interfață grafică. Execuția aplicației începe, în general, cu prezentarea unei interfețe de interacțiune cu omul (printr-un interpretor de comenzi sau a un meniu). De obicei, o aplicație este un program compilat şi stocat în forma executabilă, fiind lansat (adică încărcat în memoria de lucru şi executat pe întreg lotul de instrucțiuni) la inițiativa utilizatorului.

Structura generică a unei aplicații cuprinde două părți generice: colecțiile de date („data" - adică valori cu care se lucrează) şi *logica de prelucrare* („business logic" - adică acțiunile asupra datelor). De exemplu, o aplicație bancară conține o parte privitoare la conturi şi valorile lor, o parte privitoare la operațiuni de transfer între conturi.

4.2.2.2 Aplicații de Baze de Date

O categorie specială de aplicații sunt create şi funcționează prin intermediul Sistemelor de Gestiune a Bazelor de Date. Aceste aplicații organizează datele în structuri de tip articol (v. §2.3.5.3) care apoi sunt grupate în „tabele" ce pot fi stocate ca fişiere (pe suport extern – disc). Pentru prelucrarea datelor, se folosesc programe scrise în limbaje proprii SGBD sau în limbajul standard SQL. Programele sunt interpretate sau compilate în cod intermediar, astfel că execuția lor nu poate avea loc decât în prezența mediului de baze de date (SGBD) care este rezident în memorie şi coordonează toate acțiunile din program.
O bază de date este constituită din mai multe tabele, cu legături între ele, pe lângă care se prevăd module de prelucrare specifice aplicației vizate. Cele mai uzuale aplicații cu baze de date privesc gestiunea resurselor într-un domeniu dat: contabil, financiar-bancar, mijloace fixe, stocuri (magazii sau magazine), sisteme de vânzări, resurse umane. Multe aplicații în Internet au în fundal un server de baze de date (maşinile de căutare, magazine virtuale, etc.).

4.2.2.3 Aplicații client-server

Există aplicații în rețele de calculatoare care trebuie să asigure transferul de date şi prelucrarea acestora la distanță. Datele sunt stocate la un punct central, pe un calculator care le gestionează printr-un program denumit *server* pentru că oferă servicii (de acces şi prelucrare date), rulând în permanență şi aşteptând cereri de la utilizatori. Pe maşina locală a fiecărui utilizator, există un program denumit *client* prin intermediul căruia utilizatorul poate solicita servicii serverului distant; datele „coboară" („download") de la server pe maşina utilizatorului, unde sunt prelucrate local şi afişate (prin interfața utilizator) de către programul client. Structură de aplicație *cu două părți* este denumită, în jargonul informatic, *2-tier*.

Primele aplicații în rețea realizau toate prelucrările pe o maşina centrală (numită *gazdă*) iar maşina locală era folosită doar pentru afişare text şi preluarea datelor de la utilizator (*terminal*), adică aveau „o parte" 1-tier. Avantajele modului de lucru 2-tier provin din faptul că datele pot fi gestionate şi asigurate mai bine într-un singur punct (nu distribuite în mai multe puncte) dar prelucrările nu încarcă doar maşina centrală ci şi maşinile locale (ele fiind mai multe şi încărcate temporar). În această structură, partea client conține programele de prelucrare locală şi de prezentare a rezultatelor către utilizator, complementar părții server – care asigură prelucrări de acces şi transfer a datelor centralizate. Cele două piese software, server şi client, conlucrează şi comunică prin intermediul unui *protocol* (ca un limbaj cu set de reguli pentru formularea şi servirea cererilor), iar comunicația fizică se realizează prin infrastructura de comunicație (rețea de calculatoare). Un exemplu uzual de aplicație client-

server este WWW, în care parte server găzduieşte paginile web (şi programele de interacţiune cu utilizatorul) iar partea client o constituie navigatorul Internet („browser" ca MS Internet Explorer sau Netscape Navigator). Protocolul de comunicaţie între cele două piese software server şi client este HTTP (HyperText Transfer Protocol) şi apare indicat chiar în adresa de acces (URL) a serverului.

Pentru creşterea siguranţei accesului la date, se poate ca partea server să fie despicată în alte două părţi: una ce conţine logica de prelucrare. În acest fel, după accesul utilizatorului la maşina logica de prelucrare centrală verifică autenticitatea şi drepturile utilizatorului, apoi – doar prin intermediul programului server, oferă acces la datele aflate pe una sau mai multe servere cu date. Aplicaţii de baze de date pot funcţiona în arhitectura client-server 2-tier. În forma 3-tier funcţionează aplicaţii distribuite – cum ar fi de exemplu o aplicaţie de vânzări de acţiuni, în care datele despre cursul acţiunilor, apoi ştirile despre acţionari sau companii, respectiv tranzacţiile se găsesc pe maşini diferite.

4.2.2.4 Miniaplicaţii

O categorie specială de program o constituia miniaplicaţia (*„applet"*) – ca program ce nu există de sine stătător ci doar în cadrul unei (alte) aplicaţii – de exemplu o aplicaţie web cu pagini conţinând text şi imagini. Miniaplicaţia „applet" poate rula doar când este lansată de un eveniment extern – click  pe o imagine, producând spre exemplu animaţia unui obiect pe ecran. Miniaplicaţia „applet", ca program, este descărcată de pe maşina server pe maşina client şi rulează (de obicei în mod interpretat) pe clientul web (navigatorul Internet). Miniaplicaţia „applet"este un program script (adică scris într-un limbaj scriptural) şi interpretat linie cu line din textul sursă, acţiunile sale fiind astfel permanent controlate; din motive de securitate, miniaplicaţia „applet" nu are acces la sistemul de fişiere al maşinii client iar comunicaţia o poate realiza doar cu maşina server de pe care provine.

Complementar miniaplicaţiilor „applet" există *„servlet"*, ce rulează pe maşina server spre a asigura: acces securizat la baze de date, facilităţi pe maşini de căutare în Internet, generarea de pagini web dinamice. Şi acest tip de program se scrie în limbaj scriptural şi rulează interpretat, în cadrul unei aplicaţii gazdă (de exemplu serverul web).

4.2.3 Instrumente software de dezvoltare a aplicaţiilor

În accepţia comună, clasică, programarea constă în scrierea textului sursă într-un limbaj de programare ales, prin care se descrie un algoritm de prelucrarea a datelor. Această activitate este foarte laborioasă şi necesită dese reveniri în scrierea textului (pentru corectura sintactică, pentru optimizare sau chiar refacerea codului). Ingineria programării indică metode şi paşi sistematici prin care erorile în soluţia problemei şi în realizarea codului se reduc la minim. Pentru a obţine o productivitate rezonabilă în munca de programare sunt necesare instrumente software de asistare a programatorului. Cele mai importante categorii de instrumente sunt prezentate în cele ce urmează.

4.2.3.1 Interfeţe, biblioteci şi pachete de programare

Limbajele de programare actuale tind să fie cât mai concise – privind cuvintele cheie şi comenzile de prelucrare sau control de flux (atribuire respectiv repetiţie şi decizie). O mare parte din prelucrări sunt implementate ca subprograme sau pachete de funcţii (v. §3.2.3.1 a) şi formează *biblioteci*. Programatorul trebuie să cunoască aceste biblioteci, cu inventarul lor şi modul de utilizare a subprogramelor aferente, pe care le apelează în programul sursă; la faza de editare a legăturilor („linking" v. (III) la §4.1.1.4) ele se includ în codul obiect al programului apelant într-una din modalităţile:

Un tip de bibliotecă foarte utilizat este *Interfața de Programare a Aplicațiilor* (*API*), care conține seturi de funcții, proceduri, variabile şi structuri de date ce pot fi folosite de către programator în aplicații ca obiecte precompilate, apelate standard. Există API care fac parte din sistemul de operare sau livrate ca pachet independent.

Trebuie făcută o deosebire clară între „pachete de programe" (sau clase) – ca biblioteci şi „pachete software" sau „pachete de aplicații" – ca modalități de distribuție şi instalare software achiziționat de la producători, de exemplu pachete pentru aplicații de birou – MS Office al firmei Microsoft, sau StarOffice al firmei SUN.

4.2.3.2 Medii de programare

Dezvoltarea unui program scris într-un limbaj de programare ales, decurge mai uşor dacă toate instrumentele necesare sunt integrate într-unul singur, adică: editorul de text sursă, compilatorul, editorul de legături şi chiar depanatorul. Un *Mediu de Dezvoltare Integrat* (*IDE* – „Integrated Development Environment") este un instrument software care oferă aceste facilități, fiind utilizat atât la proiectarea cât şi la implementarea aplicației. Fiindcă interfața grafică către utilizator (GUI – „Graphical User Interface") este partea de program care necesită cel mai mare efort de programare, acestor medii li s-a adăugat modalitatea de proiectare şi implementare vizuală interfeței grafice. Prin *programarea vizuală*, se plasează obiecte (de tipul fereastră, buton grafic, casetă de text, etc.) pe o suprafață de lucru – care la final va deveni suprafața de lucru a utilizatorului, în fundal, mediul de programare producând cod prin care rezolvă partea de program ce va construi obiectele interfeței. Exemple de medii (IDE) sunt populare le Turbo C sa Turbo Pascal (ale firmei Borland), astăzi prezente şi pe sistemul de operare Linux în mod gratuit. Medii moderne, cu posibilități de lucru pentru obiecte diverse (grafice şi nu numai) sunt MS Visual Basic şi MS Visual Studio (pentru mai multe limbaje – C++, C#, Visual Basic), mediul Delphi (bazat pe limbajul Pascal), medii pentru Java.

4.2.3.3 Medii pentru Baze de Date

Pentru crearea şi utilizarea aplicațiile cu baze de date se folosesc *Sisteme de Gestiune a Bazelor de Date* (SGBD, în engleză DBMS – Data Base Management Systems). Acestea sunt medii integrate care oferă instrumente de creare a obiectelor din aplicațiile cu baze de date (v. §4.2.2.2) – ca grile de proiectare tabele sau interogări, cu mijloace vizuale pentru rapoarte şi formulare, dar şi limbaje de descriere şi manipulare a datelor – prin care se creează tabele şi se formulează interogări. SGBD este un mediu în care se creează obiectele de tip tabel, interogare, formular, raport, iar aplicația – ca set de module program scris în limbajul de manipulare a datelor, poate rula numai sub mediul SGBD (adică în prezența acestuia – încărcat în memorie).

SGBD sunt cele mai folosite de medii de implementare a Sistemelor de Informatizare – ca structuri complexe ce integrează aplicații în mai multe domenii de activitate ale unei companii (de exemplu conducere, gestiune, producție). Între produse SGBD actuale proprietare („closed source") se numără: Oracle, Informix, Microsoft Access, Microsoft SQL Server, iar cu cod liber („open source") MySQL, PostgreSQL, SQLite.

4.2.3.4 Instrumente de asistare a dezvoltării software

Complexitatea aplicațiilor şi timpul scurt disponibil pentru a le proiecta şi implementa, au impus noi modalități de dezvoltare software. Conceptul de *Dezvoltare Rapidă a Aplicațiilor* (*RAD* – Rapid Application Development) este o soluție productivă de proiectare şi implementare rapidă a aplicațiilor, bazată pe instrumente vizuale şi asistenți („wizards"). Programarea foloseşte limbaje ca de generația a 4-a (denumite 4GL - în timp ce 3GL sunt Pascal, C, COBOL) şi decurge ca o proiectare a aplicației elaborată cu ajutorul calculatorului;

codul propriu-zis este generat de către mediul RAD, în fundal. Exemplele de instrumente RAD sunt Visual Basic şi similarul lui Oracle PowerObjects, apoi C# de la Microsoft – ca limbaj, dar de fapt în întregime un instrument RAD, precum şi medii de baze de date ca CA Visual Objects sau MS Visual FoxPro. Într-un instrument RAD de tipul VisualBasic se folosesc *programarea orientată pe evenimente* („event driven programming" – prin care acțiunile din program sunt ataşate butoanelor grafice) şi componentele (v. pct. iv) la 4.2) puse la dispoziție de către mediu.

O altă categorie de instrumente eficiente sunt *instrumentele CASE* („Computer Aided Software Engineering" sau „Computer Automated System Engineering"), care oferă asistență în una sau mai multe etape de dezvoltarea a produsului (sau întregului sistem) software dar în special pentru etapele de analiză, proiectare şi testare. Există medii CASE ce permit crearea modelelor apoi arhivarea şi utilizarea lor partajată. Funcțiile instrumentelor CASE pot fi:

- Editarea sau alegerea de modele pentru interfețe utilizator grafice (GUI);
- Generarea automată de cod
- Generarea automată a documentației
- Verificarea consistenței între diagramele de proiectare şi modelele rezultate;
- Asistență pe durata întregului ciclu de viață al programului.

Un instrument CASE rulează ca un program care conduce analistul prin etapele obligatorii de dezvoltarea a produsului software, după o metodologie de analiză / proiectare aleasă din mai multe – oferite de instrument.

4.3 Sisteme de operare şi programe utilitare

Funcționarea unui calculator nu este posibilă un sistem de operare rezident – adică încărcat în memoria de lucru şi în execuție; el este denumit software de bază pentru că oferă un mediu de execuție pentru alte programe – software de aplicație.

4.3.1.1 Funcțiile ale sistemului de operare

Un sistem de operare trebuie să asigure în principal:

i) Alocarea resurselor – adică a spațiului de memorie, procesorului (sau procesoarelor), unităților de intrare / ieşire.
ii) Planificarea şi controlul programelor în execuție – adică rularea programelor după priorități şi introducerea oprirea lor temporară când acestea aşteaptă la o resursă partajată (folosită în comun cu alte programe).
iii) Interpretarea şi execuția comenzilor de la operatorul uman – adică oferirea unui mijloc de furnizare a comenzilor (ca text sau ca evenimente pe o interfață grafică) apoi conversia lor în acțiuni corespunzătoare.

Comenzile sistemului de operare se referă la transferuri de date şi programe între periferice şi chiar cu memoria de lucru caz în care transferul se spune că „se deschide" – adică se lansează în execuție colecția de date sau programul. Procesorul este alocat unui program în măsura în care alte programe mai prioritare nu îl solicită sau programul în cauză aşteaptă la un periferic încheierea unei operații de intrare sau ieşire.

În general, operarea la un sistem de calcul se poate face în două moduri:

- *regim comandă* – în care comanda se furnizează ca un cuvânt (sau o prescurtare), însoțită de parametri;
- *regim meniu* – în care comanda şi parametrii se aleg din liste şi subliste de opțiuni (meniuri).

În regimul comandă, trebuie cunoscute cuvintele ce exprimă comenzile şi parametrii, precum şi sintaxa se combinare a acestora – pentru ca o comandă furnizată să fie corect interpretată şi

executată; în plus, comanda trebuie înscrisă de la tastatură deci necesită un timp de editare (eventual şi corectură). În regimul meniu, nu este necesară cunoaşterea comenzilor ci doar recunoaşterea lor din listele de meniu, iar execuţia lor se face prin selecţia din listă a opţiunii (eventual cu sub-opţiuni) dorite; în plus, se poate oferi şi ajutor („help") pe loc; se spune că lucrînd în acest mod programul este „prietenos".

Sistemele de operarea moderne (cum sunt Windows ori LINUX şi alte sisteme UNIX) oferă facilităţi *multitasking* – de execuţie a mai multor programe (sau activităţi – „tasks") simultan, precum şi facilităţi multi-utilizator – de lucru pentru mai mulţi utilizatori simultan, cu acces prin reţele de calculatoare. Pentru funcţionarea *multiutilizator*, se creează şi menţin contexte specifice fiecăruia dar, ca şi pentru programe, fiecare este servit pe rând, rapid, astfel că aparent lucrează simultan.

4.3.1.2 Programe utilitare

Configurarea resurselor sistemului de operare (de ex. pentru periferice, reţea, limba şi ora locale), manevrele de gestionare a resurselor (vizualizare, copiere, ştergere directoare şi fişiere,) şi comenzile de lucru pentru transferul datelor între periferice şi memorie (tipărire, lansare programe, acces la reţea) sunt acţiuni dificile, care se pot executa, în general, prin comenzi complicate. Pentru a facilita aceste acţiuni, se folosesc programe *utilitare* – programe considerate între programele de bază şi programele de aplicaţii, ce asigură manevrarea comodă a setărilor şi resurselor calculatorului.

Următoarele categorii de programe utilitare sunt mai des folosite:

1. Managere de resurse – care permit vizualizarea conţinutului discurilor (fixe, flexibile, optice) şi manipularea directoarelor şi fişierelor (copiere, mutare, etc.). Exemple sunt Windows Explorer, sau Windows Commander, care prezintă în panourile de lucru ale ferestrelor acestor utilitare structura arborescentă de directoare şi fişierele, apoi permit operaţii cu acestea.
2. Arhivoare – care permit salvarea conţinutului directoarelor şi fişierelor, cu compactarea şi organizarea informaţiei pentru păstrarea sa pe suport extern un timp îndelungat. Exemple sunt: Backup/Restore Windows, WinZip, WinRAR, Ace.
3. Antiviruşi – care permit controlul şi eliminarea programelor de tip virus, vierme, cal troian, bombă logică ce pot „infecta" un sistem de calcul spre a reduce sau anula capacitatea de lucru sau a distruge date şi chiar unităţi periferice. Exemple sunt: RAV (Romanian AntiVirus – cumpărat de Microsoft), Bitdefender, MacAfee Antivirus.

Programele utilitare sunt parte a sistemelor de operare moderne, dar pot fi şi produse de la firme specializate.

4.4 Interfeţe utilizator

Interacţiunea om-maşină are atât aspecte tehnice cât şi psihologice. Aceste aspecte trebuie luate în considerare pentru a sigura lucrul eficient la aplicaţiile pe calculator.

Interfaţa cu utilizatorul reprezintă totalitatea mijloacelor prin care omul interacţionează cu maşina, uzual reprezentate de perifericele de intrare tastatură şi indicator pe ecran (mouse, track-ball, touch-pad – v. §1.3.1.5) cu perifericul de ieşire ecran şi imaginile afişate.

Realizarea unei interfeţe utilizator „prietenoase" priveşte, în principal, modul de afişare a obiectelor prin care omul interacţionează cu maşina. Aspectul de ansamblu al imaginilor pe ecranul unei aplicaţii este cunoscut drept „look-and-feel" şi se referă la caracteristicile vizuale ale interfeţei şi la comportamentul interfeţei ca răspuns la acţiunile utilizatorului. Interfeţele

utilizator actuale se bazează pe grafică (în sensul de imagine construită punct cu punct), numite de aceea Interfețe Utilizator Grafice (în engleză cu acronimul GUI). Acestea respectă câteva principii cum sunt:

- Utilizatorul controlează programul şi nu invers – adică omul acționează şi calculatorul răspunde prompt, eventual în mod personalizat (specific persoanei care lucrează).
- Utilizatorul operează direct cu datele de pe ecran, astfel ca să constate direct relația cauză-efect a acțiunilor sale. Obiectele reprezentate pe ecran sunt intuitive şi apropiate de obişnuința omului.
- Obiecte şi acțiuni similare în diferite aplicații au acelaşi mod de reprezentare grafică şi se comportă identic.
- Prezentarea pe ecran este clară (simplă, inteligibilă) şi estetică (atractivă).
- Interfața este tolerantă la greşelile utilizatorului oferind ajutor şi sugestii de rezolvare a problemelor apărute.

În continuare, se vor prezenta obiecte generice pe ecranul de afişare şi manevre uzuale efectuate cu perifericele de intrare, prin care utilizatorul interacționează cu maşina.

4.4.1 Tastatura şi mouse-ul

Tastatura, ca dispozitiv de intrare, permite atât introducerea simbolurilor de scriere (alfa-numerice) cât şi efectuarea unor acțiuni (ştergere caracter, deplasare cursor pe ecran, Enter – salt la nou rând sau execuție comandă/opțiune curentă). O manevră specială de introducere se efectuează prin apăsarea simultană a tastei Ctrl şi a unei litere alese (Ctrl-Litera) – combinația fiind ataşată unei comenzi uzuale în aplicația dată, fiind o „scurtătură" pentru execuția acesteia (altfel selectată dintr-un meniu, deci cu mai multe apăsări de taste). Spre exemplu combinația Ctrl-S reprezintă scurtătura pentru salvarea un unui document. Similar, pot fi alocate comenzi pentru scurtături din combinația Alt-Litera.

Pentru a indica cu mouse-ul, trebuie aşezată săgeata *indicator* pe obiectul ales. Manevrele ce se pot execut cu mouse-ul sunt:

1. Indicare – indicatorul atinge un obiect vizat; dacă zăboviți asupra obiectului, apare un mic mesaj (numit „tool tip") care conține o descriere a obiectului.
2. Click – butonul stânga apăsat scurt (selectează un obiect vizat).
3. Dublu click – butonul stânga apăsat rapid de două ori (deschide documentul sau lansează programul ataşat obiectului vizat).
4. Click dreapta – butonul dreapta apăsat (desfăşoară meniu contextual, adică specific locului vizat).
5. Glisare – click pe un obiect vizat şi deplasare mouse cu buton stânga menținut apăsat spre un alt loc pe ecran (mutare obiect vizat sau selectarea mai multor obiecte).

Indicatorul îşi schimbă forma funcție de obiectul indicat, spre a anunța comenzi se pot executa. De exemplu, când indicatorul se transformă intr-o mână cu degetul ridicat, obiectul indicat este o hiperlegătură, iar o săgeată cu doua capete permite redimensiunarea unui obiect.

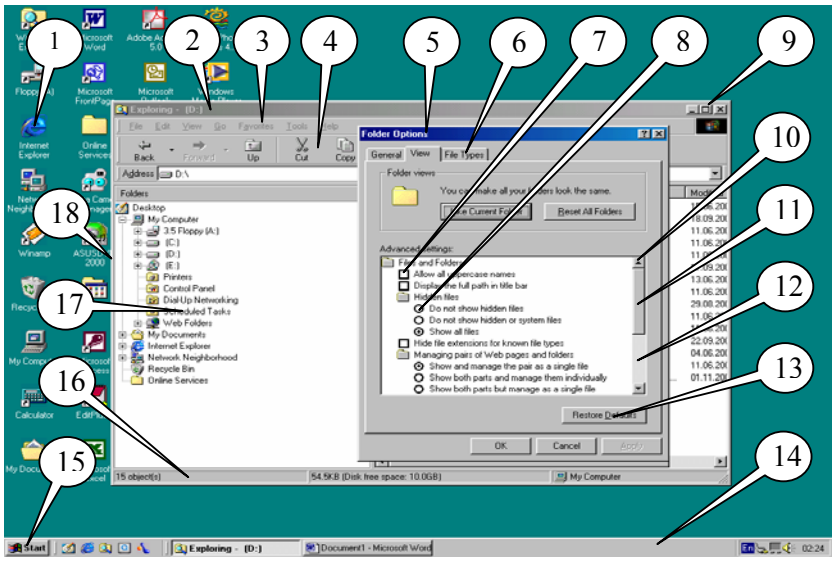
4.4.2 Ferestre şi casete de dialog

La sistemele bazate pe ferestre, utilizatorul poate comunica cu aplicația prin elementele ferestrelor şi casetelor de dialog.

4.4.2.1 Suprafața de lucru

Aşa cum tăblia mesei de lucru, la birou sau acasă, este ocupată cu obiecte de uz curent (hârtii, cărți, creioane, lampa de birou), tot astfel ecranul este ocupat obiecte utile lucrului cu

calculatorul. Fiecare din obiecte este reprezentat printr-o pictogramă (vezi (1) în figura de mai sus). Butonul „Start” (15) permite accesul la aplicațiile instalate pe mașină, precum și la opțiunile de configurare pentru periferice și pentru întregul sistem. Suprafața de lucru („desktop”) poate fi configurată, la rândul ei, prin includerea de noi obiecte sau prin înlăturarea lor. Bara de activități (14) indică aplicațiile lansate în execuție, prin butoane grafice asociate fiecăreia; aceste butoane permit activarea unei aplicații și eliminarea ei de pe ecran (minimizarea ferestrei) prin click-uri succesive pe butonul corespunzător aplicației.



Structura unei ferestre și a unei casete de dialog

4.4.2.2 Ferestre aplicație

O fereastră aplicație este o reprezentare standard a instrumentelor uzuale de lucru cu aplicația. Aceste instrumente sunt reprezentate grafic în diverse moduri și permit acționarea lor cu ajutorul mouse-ului sau al tastaturii. În tabelul de mai jos se prezintă elementele generice ale unei ferestre, rolul fiecăreia și modul său de acționare.

Element	**Mod de acționare**
Bara de titlu (2) prezintă numele aplicației, precum și numele documentului sau, în cazul navigatorului, locul vizat – disc, director pe disc, calculator din rețeaua locală („Network Neighborhood”).	glisarea → *Barei de titlu* mută fereastra pe suprafața de lucru .
Bara de meniu (3) cuprinde categorii de opțiuni accesibile în aplicația dată, din care trei sunt regăsite la orice aplicație: File (manipularea datelor specifice aplicației), Edit (copiere, căutare, ștergere) și Help (ajutor imediat – cu un cuprins, index și căutare).	click sau Alt-Litera dă lista acțiunilor specifice aplicației și datelor sale
Bara de unele (4) cuprinde comenzi uzuale ilustrate prin desene intuitive, acționate ca *butoane grafice*. Comenzi de editare, lucru cu fișiere și cu ferestre au o replică în bara de unele.	click buton grafic execută comanda (ex. = decupare)
Butoane deschis / închis (9) aflate în colțul dreapta sus, permit punerea în așteptare a aplicației (minimizarea), ocuparea întregului ecran activ (maximizarea) sau restaurarea ferestrei pentru redimensionare, iar ultimul închiderea aplicației	minimizare maximizare restaurare închidere

Bara de stare (16) prezintă informații cu setări specifice aplicației	numai vizionare
Panouri de lucru (17) prezintă datele specifice aplicației și permite manipularea lor conform metodelor specifice acestora.	prelucrare specifică a datelor aplicației
Chenarul (18) permite redimensionare ferestrei (lățire, lungire și reciproce, respectiv mărire / micșorare proporțională).	glisare chenar sau colțul dreapta-jos

4.4.2.3 Casete de dialog

Prin casetele de dialog utilizatorul poate viziona setări curente ale aplicației și poate modifica aceste setări prin acționarea elementelor casetei, după cum se prezintă în.

Element	**Mod de acționare**
Bara de titlu (5) prezintă numele casetei de dialog, (de exemplu opțiunile fișierelor – „Folder Options", în figură).	glisarea *Barei de titlu* mută caseta
Fișă (6) prezintă setări posibile, de obicei împărțite pe cadre (secțiuni tematice) – în figură tipuri de fișiere („File Types")	click pe etichetă selectează fișa
Casete de bifă (7) permit selectarea mai multor opțiuni din set dat	/„spațiu" bifează ☑
Butoane radio (8) permit selectarea unei singure opțiuni din set dat	/„spațiu" indică ⊙
Butoane de derulare pas cu pas (10) permit derularea informației din panou în fereastra de vizitare (când aceasta este neîncăpătoare)	click derulează în sensul dorit ,
Butoane de derulare continuă (11) permit derularea în mod continuu a informației în fereastra de vizitare	glisare derulează conținut
Bara de derulare (12) permit derularea pagină cu pagină a informației în fereastra de afișare a panoului	click pe zona liberă derulează o pagină
Butoane grafice (13) permit efectuarea de acțiuni specifice (înscrise pe buton); uzuale sunt „OK"- confirmare și „Cancel" – abandon.	click – execută ! (ex. OK)
Listă derulabilă – care prezintă un set de facilități ce pot fi selectate după derularea listei text până la reperul dorit.	derulează cu click pe reper – alege (ex. Always / Never / Always / When selected)
Casetă de text – care permite înscrierea unui text necesar în contextul dialogului.	introducere text dorit (ex. Initials: AV)

4.5 Rezumat

Realizarea programelor nu priveşte doar activitatea de programare (care constă în codificarea datelor şi acţiunilor) ci implică şi activităţi de analiză şi proiectare, apoi de testare şi întreţinere a aplicaţiilor realizate. Acestea sunt etape în ciclul de viaţă al produselor program, care se succed ciclic pentru că obţinerea unui produs de calitate necesită reluarea acţiunilor de analiză, proiectare dacă la testare unele părţi nu satisfac cerinţele impuse. Realizarea programelor se face astăzi utilizând medii de dezvoltare care integrează unelte de editare a programelor, de compilare şi depanare a lor, sau chiar permit generarea automată de cod după o etapă de proiectare realizată simplu, vizual (ca de exemplu prin RAD sau CASE). Limbajele de programare sunt diverse (cum sunt C, Java, SQL) şi se folosesc fiecare domenii şi contexte de dezvoltare specifice (de exemplu limbajul C pentru aplicaţii strâns legate de hardware şi sistemul de operare, Java pentru aplicaţii distribuite – mai ales prin Internet, SQL în aplicaţii cu baze de date), fiind caracterizate de nivelul apropierii de limbajul uman. Ingineria programării permite o abordare sistematică de realizare a programelor, în mod eficient şi fără greşeli constisitoare, care permite totodată organizarea dezvoltării software în mod industrial. Capitolul prezintă în ultma parte noţiuni privind sisteme de operare şi de utilizare a interfeţelor utilizator grafice actuale.

4.6 Teme de control

1. Prezentaţi comparativ conceptele de program, aplicaţie şi sistem informatic.
2. Ce persoane (ca funcţie şi rol) sunt implicate în analiza şi proiectarea aplicaţiilor?
3. Ce traseu de etape din ciclului de viaţă al unui program se parcurg dacă cerinţele formulate la prima etapă se dovedesc neîndeplinite la etapa de testare a programului?
4. Faceţi un inventare de documente întocmite de-a lungul ciclului de viaţă al unui program.
5. Ce rol au metodologiile de analiză a problemei ce se doreşte rezolvată cu un program? Daţi exemple de asemenea metodologii.
6. Ce părţi şi cu ce roluri întâlnim la o aplicaţie client-server 2-tier? Daţi un exemplu comun.
7. Ce este o miniaplicaţie şi unde este utilizată în mod curent?
8. În ce constă o bibliotecă asociată unui limbaj de programare? Daţi un exemplu.
9. Ce aduc în plus instrumentele CASE faţă de mediile de dezvoltare integrată a progamelor?
10. Ce resurse ale sistemului de calcul administrează sistemul de operare?
11. Ce utilităţi pot avea în general programele utlitare?
12. Cum credeţi că avantajează pe utilizatorul obişnuit principiile ce stau la baza interfeţelor utilizator grafice? Analizaţi două dintre aceste principii.
13. Ce manevre se pot executa cu dispozitivul indicator (mouse) şi ce scop pot avea fiecare?
14. Indicaţi trei elemente din casete de dialog şi specificaţi rolul lor.

Exemple de răspuns:

(3) Dacă nu sunt îndeplinite la testare cerinţele formulate la prima etapă, este necesară parcurgerea din nou a etapelor de Analiză, Proiectare, Implementare şi Testare, care se repetă ciclic până ce toate cerinţele sunt îndeplinite.

(9) Faţă de mediile de dezvoltare integrată a progamelor, instrumentele CASE aduc în plus: posibilităţi de generare atomată a codului şi documentaţiei, verificarea consistenţei între diagramele de proiectare şi modele, asistenţă pe durata întregului ciclu de viaţă a programului.

5 Utilizarea calculatoarelor pentru aplicații de birou

Programele care permit realizarea unei utilități umane (în prelucrarea informațiilor) se numesc *programe de aplicație*. De fapt, acestea sunt scopul construirii şi programării calculatoarelor, între cele mai uzuale aplicații putând fi enumerate (cu caracteristicile lor esențiale):

- *Procesoare de texte* – permit editarea, formatarea, corectarea şi tipărirea textelor;
- *Foi de calcul tabelar* – permit crearea de tabele, cu expresii de calcul şi funcții diverse, cu facilități de creare şi modificare histograme;
- *Aplicații de grafică* pe calculator – permit crearea, prelucrarea şi tipărirea graficii.
- *Navigatoare web* – permit accesarea site-urilor distante şi interacțiunea cu ele.
- *Poştă electronică* – permite transmiterea de mesaje şi documente la persoane fizice şi instituții care au o adresă Internet.

Între aceste categorii de aplicații, primele două şi ultimele două fac parte, în general, din pachete de tip „Office" (aplicații de birou), produse de diferite firme (de ex. Microsoft, SUN).

Capitolul 5 şi 6 au ca scop prezentarea sistematică a modului cum se organizează şi apoi cum se prelucrează informația pentru primele şi respectiv ultimele două categorii de aplicații. Acest scop se înscrie, de altfel, în ideea generală al acestei lucrări: introducere în Tehnologia Informației şi Comunicațiilor (TIC). Astfel, utilizarea pentru scopuri din viața obişnuită a uneltei numită calculator face parte în mod natural din prezentarea propusă, iar capitolele furnizează indicații de utilizare eficientă a acestor aplicații necesare şi la îndemâna oricui.

Pe parcursul acestui capitol se vor prezenta principii de lucru şi facilități oferite în general de procesoarele de texte şi de calculul tabelar. În anexe se prezintă efectiv manevre de operare pentru cele mai importante acțiuni desfăşurate la realizarea, modificarea şi afişarea documentelor la aceste aplicații, folosind produse din pachetul Office al firmei Microsoft. Totuşi, la prezentarea principiilor de lucru şi elementelor celor două categorii de produse, se vor folosi capturi de ecran din produsele amintite dar, pentru a păstra caracterul de generalitate impus de ideea acestei lucrări, se vor descrie operațiuni şi facilități ce se regăsesc la cele mai multe produse de acest fel, de la diverşi producători.

5.1 Procesoare de texte

Realizarea şi modificarea documentelor scrise este o muncă migăloasă şi dificilă, fiindcă ea implică nu doar scrierea conținutului de text, ci şi:

1. organizarea ideilor – pe capitole, subcapitole şi paragrafe,
2. modificarea textului – cu mutare a unor părți, înlocuirea sau eliminarea lor,
3. crearea de liste şi tabele,
4. inserarea de imagini, grafice, desene intuitive (cum sunt diagramele)
5. aranjarea textului în pagină, cu antet, subsol, note de subsol, numerotare de pagini
6. alegerea tipurilor de caractere, cu dimensiuni, stiluri şi efecte adecvate unui context dat
7. corectarea – ortografică şi gramaticală, a textului

Dacă unele operații (primele trei) se execută manual, pe foaia de hârtie scrisă de mână, doar apoi aplicarea operațiilor următoare (la tipografie de exemplu), realizarea documentului durează foarte mult şi nu este sigur că va arăta aşa cum ne-am dorit.

Folosind un procesor de texte, se poate face nu numai editare textului (adică scrierea şi operațiile 2, 3), ci toate operațiile de mai sus, dar şi mai mult:

8. evidența corecturilor în versiuni succesive ale documentului,

9. numerotarea automată a figurilor, tabelelor şi notelor de subsol, eventual cu indarea unei liste inventar de figuri şi tabele,
10. trimiteri la capitole, figuri, paragrafe – ce pot fi utilizate direct în forma electronică,
11. realizarea de legături cu alte documente.

În plus, documentul rezultat poate fi vizualizat, înainte de tipărire, aşa cum va arăta el la final, poate fi tipărit apoi pe foi față-verso.

În acest subcapitol se vor prezenta elemente de organizare şi formatare a documentelor scrise (cu referire la puntele 1..11 de mai sus) şi se vor evidenția modalitățile pe care un procesor de texte le pune la dispoziția utilizatorului pentru a realiza documente cu aspect profesionist. Exemplificarea acestor aspecte se va face cu pictograme din interfața procesorului de texte MS Word (fără referire la vre-o variantă), dar aspectele sunt similare şi la alte procesoare de texte (de la alți producători), unele folosind chiar aceleaşi pictograme.

5.1.1 Structura a unui document scris

În activitatea sa omul creează, modifică şi foloseşte multe tipuri de documente – funcție de profesia şi scopurile fiecăruia începând de la scrisori până la teze de doctorat. Realizarea documentului, oricare ar fi el, prezintă o *structură a informației*, o *structură a foii tipărite* şi o *structură a conținutului de idei*. Pentru fiecare din aceste structuri, procesorul de texte oferă mijloace specifice de realizare, personalizare şi automatizare a operațiunilor de editare a informației.

5.1.1.1 Structura informației în document

În continuare, se va prezenta structura de blocuri cu informații dintr-un document generic de tip *raport* care, dependent de destinația sa, poate conține sau nu elemente mai speciale, prezentate în lista de mai jos (cum sunt c., d., e.) dar prezintă obligatoriu conținut (b.). Fiecare din aceste *blocuri de informații* (codificate prescurtat prin trei litere) cuprinde elemente specifice denumite *câmpuri* („fields” – codificate prescurtat prin două litere) pe care procesorul de texte le gestionează şi ajută astfel la modificarea şi consultarea informației – în perioada editării şi apoi a utilizării ei. Blocurile, apoi elementele lor, se prezintă mai jos.

a. *Cuprins* – inventarul denumirilor de capitole şi subcapitole, cu paginile de început.
b. *Conținut* – text în care sunt inserate imagini, tabele, figuri, etc. tratând ideile de interes.
c. *Index* – inventarul cuvintelor cheie ce apar în text, cu paginile în care sunt explicate.
d. *Listă de figuri* – inventarul figurilor, cu numărul de ordine şi textul lămuritor.
e. *Listă de table* – inventarul tabelelor, cu numărul de ordine şi textul lămuritor.
f. *Bibliografie* – inventarul lucrărilor citate în textul documentului.

Cuprinsul este realizat prin intermediul *denumirilor de capitol şi subcapitol* („Heading”), etc. fiecare din acestea fiind declarate pe nivelul ierarhic corespunzător *Denumire (de nivel) 1* – capitol, *Denumire (de nivel) 2* – subcapitol, *Denumire (de nivel) 3* – subcapitol inferior, etc.). Aceste denumiri sunt declarate la editare şi colectate apoi automat la realizarea cuprinsului („Table of Contents” - TOC).

Indexul este realizat prin declararea, la editare, a *cuvintelor cheie* („index entry” - XE) care vor face parte apoi din index; ele sunt colectate automat şi introduse în lista index („index”).

Lista de figuri şi lista de tabele se realizează automat dacă anterior s-au declarat şi *figuri cu număr de secvență automat* şi *tabele cu număr de secvență automat* („sequence - SEQ”). Evidența figurilor şi tabelelor este ținută de procesorul de texte care, la inserarea unei noi figuri (sau tabel) între altele două existente permite actualizarea şi menținerea ordinii corecte ale acestor elemente.

Bibliografia se realizează prin mijlocirea *referinţelor bibliografice* („reference”, „Index and Tables” - câmp TA) introduse în text, iar apoi colectate şi legate de titlurile cărţilor respective înscrise în bibliografie („Tabel of Authorities” - TOA).

În general, un element cu legătura se numeşte referinţă („reference”) şi se declară specific (prin meniul Insert Reference – v. §5.1.6.2 vi). Unor elemente cu legătură, folosite în blocuri de informaţii, li se ataşează unui câmp prin „captură” („caption”); captura se poate face d exemplu către: o figură, un tabel, un denumire de capitol sau subcapitol („Heading”), notă de subsol („footnote”) sau relaţie de tip ecuaţie („equation” - EQ).

5.1.1.2 Structura de conţinut a documentului

Blocul cu informaţii esenţial al unui document este *conţinutul* său. Acesta cuprinde – pe lângă text, imagini, tabele, liste, etc., ca forme de reprezentare a informaţiei. Elementele de structură ale conţinutului sunt însă:

a. *Denumire de capitol sau subcapitol* („Heading”) – exprimare generică impersonală a ideilor din capitol sau subcapitol, care nu se încheie cu punct (chiar propoziţie fiind).
b. *Secţiune* – parte din conţinut separată prin marcaje speciale (separator de secţiune – „section break”) şi de capetele de document (început sau sfârşit).
c. *Pagini* – parte de conţinut tipărită pe o singură foaie, separată de restul conţinutului prin curgerea naturală a textului sau prin separator de pagină („page break”).
d. *Coloană* – aranjare a textului pe verticală, de obicei pe o lăţime mai mică decât lăţimea paginii, pentru înlesnirea citirii rapide; este declarată prin coloane multiple („column”).
e. *Paragraf* – parte de text, formată dintr-una sau mai multe fraze, pe una sau mai multe linii de text, ce se referă la o singură idee sau un singur obiect („paragraph”).
f. *Notă de subsol* – informaţie lămuritoare numerotată, înscrisă în josul pagini (şi separată, în general, cu o linie de restul textului). Conţine referinţe bibliografice, trimiteri sau note ce detaliază aspecte particulare din text.

Fiecare din aceste elemente trebuie declarate specific: denumirile prin elemente de stil (v. §5.1.6.1), secţiunile şi paginile prin separatori inseraţi în conţinut, coloanele prin aranjarea textului folosind opţiuni „column”, paragrafele ca text unitar înscris fără salt la nouă linie („Enter” sau „Return”), notele de subsol prin inserarea indicelui superior („footnote”).

5.1.1.3 Structura foii tipărite

Fiind destinat a fi tipărit pe hârtie, documentul trebuie pregătit pentru a fi cuprins corect pe foaia de hârtie. În structura foii tipărite există:

a. *Margini libere* („Margins”) – la bordurile sus, jos, stânga, dreapta ale foii. În ceste borduri nu este recomandat să se înscrie text (v. §5.1.2.2, fiindcă procesorul permite totuşi acest lucru, dar nu mai are sens declararea marginii libere).
b. *Rigolă* („Gutter”) – bordură laterală (stânga – pe pagini impare, dreapta - pe pagini pare) lăsată liberă pentru legarea foilor documentului într-o carte.
c. *Antet* („Header”) – zonă de text pe marginea de sus a foii, în care se înscriu, de obicei, informaţii referitoare la emitentul documentului: numele firmei sau autorului, logo (mică imagine simbol al firmei), adresa, numere de telefon, numărul de pagină (uneori).
d. *Subsol* („piciorul” paginii – „footer”) – zonă de text pe marginea de sus a foii, în care se înscriu, de obicei, informaţii referitoare la versiunea documentului, data şi ora emiterii, texte predefinite („autotext”), numărul de pagină (eventual „din numărul total” – de ex. 5/22) plasat în extrema dreaptă sau în centrul paginii pe bordura de jos.

Aceste elemente se stabilesc din opţiunea referitoare la „încadrarea în pagină” („Page Setup”), iar pentru antet şi subsol din opţiunea de vizionare a documentului („View”).

La tipărirea documentului pe foi față-verso, rigola se va plasa diferit pe o față și pe cealaltă – în oglindă („mirror margins”). Similar, pentru antet și subsol se pot stabili texte diferite pentru față și pentru verso (pagină pară și impară) sau diferite doar pentru prima pagină din capitol.

5.1.2 Scrierea textului – caractere și paragrafe

Textul propriu-zis este o înșiruire de *simboluri de scriere* grupate în *cuvinte*, acestea în *fraze* pe *linii*, formând *paragrafe*. Aceste banalități a fost amintite aici pentru că elementele menționate constituie elementele de bază pentru prelucrarea textelor, efectuate de procesorul de text. Astfel, cuvintele trebuie separate cu un singur spațiu (obligatoriu chiar și atunci când apar semne de punctuație) pentru ca procesorul să poată executa analizele ortografice și gramaticale corect dar și pentru respectarea regulilor estetice și ortografice uzuale.

Selectarea (marcarea) zonelor de text prezintă comenzi speciale pentru fiecare din elementele menționate: dublu click[2] – selectează cuvântul indicat, triplu click[3] – selectează paragraful, indicator pe marginea stânga a paginii (ia forma) și click – selectează rândul indicat, etc.

5.1.2.1 Caractere

Literele, cifrele, semnele de punctuație și alte semne grafice, folosite în text, pot avea diferite desene și moduri plasare pe foaie și moduri de evidențiere. Acestea sunt

a. *Tipul caracterului* („font”) – desenul literei (simbolului) de scriere, cu un nume: Times New Roman, Arial, Courier New, **Impact**, Book Antiqua, Broadway, Σιμβολ, Verdana, University Roman, La Bamba ,♦♑■♎♑■♈♦ (Windings).

b. *Stilul caracterului* („font style”) – evidențierea caracterelor într-unul din modurile: **Îngroșat (aldin – „bold”)**, *Înclinat (cursiv - „italic”)*, Subliniat („underline”).

c. *Dimensiunea caracterului* – mărimea sa exprimată în puncte tipografice (pt.):

8 pt, 10 pt, 14 pt, 18 pt, 28 pt, 36 pt.

d. *Efecte* – indice^{superior} , indice$_{inferior}$, MAJUSCULE MICI, Umbrit, Sculptat, ~~Tăiat~~

e. *Culori* – Roșu, Verde, Albastru, Gri.

f. *Spațiul între caractere*: Normal, E x p a n a d a t , Condensat

g. *Poziția pe rând*: pe Mijloc, Ridicat, Coborât

h. *Animație*: Clipire, „Las Vegas Lights” (firme luminoase), „Sparkle Text” (Artificii)

Aceste facilități ale caracterelor se utilizează cu moderație, pentru a nu încărca textul cu forme prea multe și inutile. De exemplu, nu se evidențiază un text cu mai multe metode simultan – de exemplu ***îngroșat-subliniat-înclinat***. Dimensiunile caracterelor se aleg funcție de partea din text scrisă: Denumire capitol, Denumire subcapitol, **Alineat**.

5.1.2.2 Paragrafe

Pe parcursul unui paragraf saltul la rând nou se face automat, astfel ca utilizatorul să nu fie preocupat de formă ci numai de conținut. Sfârșitul de paragraf este ocupat de simbolul ¶, care este și depozitarul întregii informații de formatare a paragrafului. Acest simbol și altele – asociate nu unor caractere ci unor acțiuni în text, se fac vizibile prin apăsarea butonului ¶. Asemenea acțiuni, pe lângă saltul al nou rând, sunt saltul al nouă pagină, la nouă secțiune, etc.

Paragrafele pot fi:

a. *Aliniate* (față de marginile libere ale foii) – stânga, dreapta, centrat sau justificat („justify”) - marginile foii sunt şi verticalele de aliniere ale rândurilor
b. *Indentate* – adică adâncite spre interiorul paginii, dinspre stânga sau dinspre dreapta. Cel mai uzual mod de adâncire este al primei linii din paragraf, spre dreapta („First line indent”). Modificarea adâncirilor se poate face şi din marcaje pe rigla orizontală.
c. *Spațiate* – la un rând („Single”), la un rând-jumate (1/2 Lines), la două rânduri („Double”).
d. *Separate cu spații libere* înainte sau/şi după paragraf („After”, „Before”).

Alinierea paragrafelor se face numai cu ajutorul opțiunilor de meniu („Paragraph”) sau din butoanele aflate în din bara de unelte prin  pe unul din butoanele  .

Indentarea se realizează din opțiunile de meniu („Paragraph”) sau din marcajele de pe rigla orizontală: cap săgeată „Left indent”, cu pătrățel „Hanging indent”, „First Line indent”) şi „Right indent”. Indentarea cu un spațiu predefinit se poate realiza şi cu butoane pe bara de unelte:  - „Increase indent” şi  - „Decrease indent”.

Alinierea sau indentarea folosind spații repetate sau tabulatori face din calculator o maşină de scris, deci nu contribuie de loc la eficiența muncii de editare ba mai mult încurcă, iar investiția într-un calculator nu se justifică.

> Paragrafele pot fi încadrate în chenare – atunci când se doreşte scoaterea lor în evidență, tratând o idee foarte importantă. Încadrarea în chenar se poate face şi pentru întreaga pagină. Chenarele pot avea efecte precum umbre, şerpuiri, culori.

Paragrafele care încheie o pagină pot să aibă primul rând pe o pagină şi restul pe cealaltă (paragraf văduv) sau ultimul rând să fie singur pe pagina nouă (rând orfan). Fiindcă acest lucru nu este estetic, se poate seta o opțiune de „paragraf fără rând văduv/orfan” („Widow/Orphan control”) prin care liniile se mențin grupate (adică) pe aceeaşi pagină şi se lasă rânduri libere pe pagina precedentă.

5.1.3 Liste şi tabele

Aranjarea textului pe linii marcate (în *liste*) şi în plus folosind coloane (în *tabele*) permite observarea mai uşoară a clasificărilor sau ordinii între idei şi obiecte. Listele şi tabelele sunt deci forme de organizare vizuală a mai multor obiecte, în cazul tabelelor incluzând şi proprietățile lor. Utilizarea listelor şi tabelelor este indicată atunci când se enumeră obiecte sau idei diferite (şi proprietățile lor), când se face o clasificare sau când se urmăreşte o organizare clară a multor informații diferite privitoare la un set de obiecte.

5.1.3.1 Liste numerotate şi ne-numerotate

O listă de idei este mai uşor lizibilă decât un text ne-organizat, fiindcă prezintă o fiecare idee constituie un *reper* în listă şi poate fi marcat cu diverse simboluri.

Dacă informația de bază a reperelor are caracter de succesiune, cronologic sau ierarhic, ordinea trebuie subliniată cu *simboluri ordinale* (care dețin ordine intrinsecă), cum sunt numerele (în scriere arabă sau romană) sau literele (majuscule sau nu).

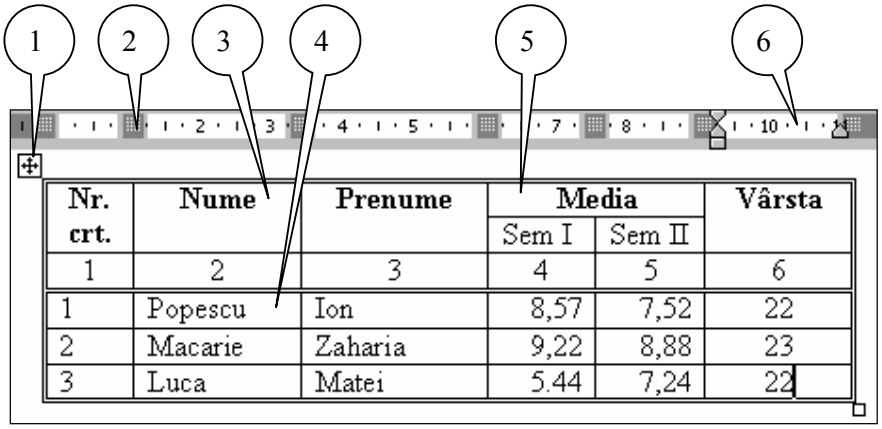
O listă marcată cu simboluri ordinale se numeşte *listă numerotată* („numbered list”). O listă ierarhizată este *listă este numerotată cu subpuncte* („outline numbered list”). Dacă informația de bază nu necesită prezentarea ordonată reperele se marchează cu simboluri grafice într-o

listă ne-numerotată (cu buline – „bulleted list”). Lista poate prezenta la fiecare reper categorii (adică este ierarhizată) şi devine *listă ne-numerotată cu subpuncte*).

Listele pot fi create chiar în timpul editării sau la finalul creării ei, prin declararea tipului de listă prin opțiunea de meniu („Bullets and Numbering”) sau prin butoanele . Un subpunct se obține prin adâncirea textului cu Tab la început de rând sau cu - „Increase indent”, iar revenirea la nivelul subpunct anterior se face cu Shift-Tab la început de rând sau cu - „Decrease indent”.

5.1.3.2 Tabele

Organizarea informațiilor în tabele este utilă pentru date structurate de tip articol (v. §2.3.5.3).



Nr. crt.	Nume	Prenume	Media		Vârsta
			Sem I	Sem II	
1	2	3	4	5	6
1	Popescu	Ion	8,57	7,52	22
2	Macarie	Zaharia	9,22	8,88	23
3	Luca	Matei	5.44	7,24	22

Structura unui tabel şi rigla orizontală asociată.

Pentru tabelul prezentat există următoarele elemente şi modalități de lucru:

(1) *marcaj tabel* prin intermediul căruia se poate selecta şi muta întreg tabelul (diametral opus este *marcajul de redimensionare*).

(2) marcaj *separator de coloană* în rigla orizontală – permite modificarea dimensiunii coloanei, prin glisare . Marcaje similare există pe rigla verticală – pentru redimensionarea rândurilor.

(3) *coloană* (rubrică) în care se află *celule* (4), cu valori înscrise.

(5) celulă comasată (opțiunea „Table-Merge Cells”) pentru coloane cu subcoloane. Operația inversă, de despicare a celulei, se face prin opțiunea „Table-Split Cells”).

(6) este spațiul unei celule care prezintă (la editarea sa) marcaje specifice de indentare.

Modificarea dimensiunilor coloanelor şi rândurilor se poate face şi prin glisare , când indicatorul este plasat pe liniatură şi ia forma - pentru coloane sau - pentru rânduri.

5.1.3.3 Imagini şi plasarea lor în text

Imaginile care se pot introduce într-un document scris pot fi aduse din biblioteca de clip-uri ataşată produsului („Clip Art”) sau din fişiere aflate pe disc („From File”), ambele posibilități fiind accesibile din meniu („Insert - Picture”) sau din bara de unelte.

După inserare, se pot stabili diferite proprietăți ale acestor obiecte (meniul „Format Object”):

a. *Culorile fondului şi liniei, tipul de linie*(„Colors and Lines”);

b. *Dimensiunea imaginii* („Size”) :poate fi redimensionată la scară dată sau exprimată dimensiunea explicit în unități de măsură – țoli sau centimetri;

c. Plasarea imaginii în text („Layout”) - indicată sugestiv prin pictograme cu imaginea înconjurată de text (într-un cadru depărtat „Square”, cadru strâns „Tight” sau direct înconjurată „Through”), în spate („Behind text”) sau peste text („In front of text”), separată de text („Top and Bottom”) ori în text („In line with text”)

Imaginile sunt obiecte plasate în document, care pot fi redimensionate (indicator de mouse plasta pe muchia sa, devine ↔ apoi este glisat ), pot fi copiate şi mutate (prin manevre de editare), pot fi transferate între aplicații (de prelucrare grafică, de calcul electronic, etc.)

5.1.4 Verificarea şi corectarea ortografică şi gramaticală.

Scriitorii de meserie foloseau maşina de scris direct în momentul creației. În mod similar, astăzi, mulți folosesc calculatorul. În momentele în care ideea se aşterne pe hârtie (fie ea şi electronică pe ecranul calculatorului) nu este posibil, şi nu este indicat ca atenția creatorului să fie îndreptată spre alte chestiuni decât ideea. De aceea, este posibil să apară erori ortografice (adică privind corectitudinea cuvintelor) şi gramaticale (adică privind modul de combinare a cuvintelor). Procesorul de texte ajută şi în acest caz, realizând verificarea ortografică şi gramaticală, apoi indicarea cuvintelor şi frazelor eronate sau chiar corectarea celor la care „recunoaşte” erori sistematice (de exemplu inversarea unor litere într-un anumit cuvânt).

Verificarea ortografică şi gramaticală trebuie „activată” din meniu corespunzător („Tools – Options - Spelling and Grammar – Check spelling as you type” şi Check grammar as you type”). Erorile ortografice apar *subliniate cu linie şerpuită roşie*, iar cele gramaticale *subliniate cu linie şerpuită verde*.

Unele cuvinte se pot corecta automat dacă au fost înscrise cu forma eronată – ca eroare sistematică (în coloana „Replace”) şi cea corectă (în coloana „With”) în lista „Replace text as you type” oferit de Corectorul Automat („AutoCorrect”). Corectarea se face folosind tot ajutorul procesorului, astfel: *corectura ortografică*: pentru cuvintele subliniate se pot solicita sugestii privind sinonime sau antonime preluate intr-un dicționar de cuvinte instalat alături de procesor. Cuvintele sinonime se pot consulta şi cu ajutorul opțiunii Tezaur („Thesaurus”).

5.1.5 Lucrul cu documente

Crearea textului şi modificarea sa necesită operații multiple, cum sunt de exemplu copiere sau mutarea unor zone de text, căutarea unor şiruri de caractere (frânturi de cuvinte, cuvinte sau chiar fraze) şi, eventual, înlocuirea lor cu şiruri de caractere. După ce textul a fost realizat este indicată salvarea lui în forma curentă (chiar ne-finalizată), pentru ca lucrul să fie reluat mai târziu, pentru utilizarea formei electronice sau pentru tipărire ulterioară. Aceste chestiuni se discută pe scurt în continuare.

5.1.5.1 Manevre de Editare

O modificare în text cuprinde, de obicei, trei faze:

1. *Selectarea zonei* text de editat – prin care se face marcarea (indicarea) zonei de editat, cu manevre descrise la începutul §5.1.2; la acestea se adaugă: glisare  peste zonă, parcurgerea ei cu Shift-săgeți (taste →, ↑, ↓, ←).
2. *Manevra pregătitoare* – prin care se face o copie a zonei de editat într-o memorie anume („Clipboard”); de exemplu comanda „Edit – Copy”  sau „Edit – Cut”  .
3. *Manevra de modificare* – prin care se face efectiv schimbarea dorită; de exemplu mutarea cursorului clipitor şi comanda „Edit - Paste”  , schimbare font, etc.

5.1.5.2 Lucrul cu ferestre

Pentru operațiuni efectuate cu mai multe documente, ferestrele acestora se pot aranja pentru a înlesni vizionarea lor pe acelaşi ecran, fără a comuta de pe una pe alta. Se pot vizualiza mai multe ferestre prin opțiunea „Arrange All", dar se pot vizualiza două părți ale aceluiaşi document simultan, prin opțiunea „Split" – care „despică" documentul şi permite derularea independentă a fiecărei părți, pentru vizionarea lor simultană (de exemplu atunci când se doreşte consultarea uneia din părți în timp ce se lucrează la cealaltă).

5.1.5.3 Şabloane şi scrisori tip

Pentru documente uzuale, procesorul de text pune la dispoziție şabloane („Template") care vor fi doar completate cu conținutul dorit de utilizator. Forma şablonului este preluată din biblioteca de şabloane, uzual fiind oferite şabloane pentru scrisori, fax-uri, CV-uri, Agendă.

Pentru realizarea de scrisori tip ce trebuie transmise la mai mulți destinatari, acestea se pot personaliza folosind *Asistentul de scrisori multiple* – „Mail merge Wizard" (meniu „Tools"). Astfel, se poate „construi" un text care să conțină şi o structură de câmpuri în care se vor înscrie automat informații dintr-un tabel cu date asupra persoanelor destinatare: apelativul (Dl., D-nă, Dr., etc.) numele şi prenumele, adresa, şi altele. În final, scrisoarea tip va afişa în locul câmpurilor datele concrete şi se va putea tipări; se parcurge printr-o opțiune de meniu (sau buton) lista de persoane, tipărindu-se pentru fiecare exemplarul dedicat. În acest mod se îmbină date (despre persoane) şi documente (scrisori) prin procesorul de texte.

5.1.6 Indicații de tehnoredactare computerizată

Procesorul de texte pune la dispoziția utilizatorului toate mijloacele pentru crearea de documente cu aspect profesionist care, în plus, pot fi modificate uşor sau pot fi elaborate în echipă. Pentru a utiliza corect şi complet un procesor de texte, este bine ca la crearea documentului să se respecte recomandările făcute în acest subcapitol.

5.1.6.1 Definirea stilurilor de formatare

Funcție de destinația documentului se stabileşte o mulțime de stiluri dorite (sau impuse) pentru structura şi modul cum vor apare pe hârtie. Lista de stiluri disponibile (accesată prin ) este predefinită dar se poate completa cu altele după trebuință. Stilurile de formatare se referă la:

a. Casractere – privind caracteristici precum: tipul, dimensiunea, stilul caracterului pentru diferite părți ale documentului (denumiri de capitol – „Headings", textul „Normal", note de subsol „Footnotes", etc.).

b. Paragraf – aliniere, indentare, stop-uri Tab, spațiul între linii, chenare, caracteristici ale literelor din paragraf. În acest context se definesc efectiv denumirile de capitol („Headings") indicând, spre exemplu, că denumirea de nivel 1 apare pe pagină nouă (v. final §5.1.2.2).

c. Tabel – privind tipul de chenar, umbriri pe coloane, culori.

d. Listă – privind modul de numerotare şi indentare a linilor, pentru diferite tipuri de liste (numerotate / ne-numerotate) – v. §5.1.3.1.

e. Figură – numele generic (de exemplu prescurtat Fig.) şi tipul de numerotare automată.

f. Tabel - numele generic şi tipul de numerotare automantă,

g. Cuprins – modul cum va fi indicat inventarul denumirilor de capitole şi pagini de start.

5.1.6.2 Paşi în redactarea corectă şi eficientă a documentului

i) Se realizează structura de denumiri de capitol pe niveluri ale documentului, fiecare declarat de tip „Heading”1, 2, etc.(v. §5.1.1.2). Această structură va fi vizibilă prin „Document map”  .

ii) Se înscriu informaţiile de antet şi subsol (logo, texte lămuritoare), eventual antet diferit pe pagină pară şi impară, numere de pagină (cu număr total sau nu), data emiterii documentului, etc.; numerotarea paginilor se poate face şi în antet.

iii) Se folosesc corect semnele de punctuaţie: spaţiu obligatoriu după semnul de punctuaţie dar niciodată înainte – excepţie parantezele ‘(, şi ,)’.

iv) Alinierea şi indentarea paragrafelor se face doar utilizând mijloacele oferite de procesorul de texte (v. §5.1.2.2).

v) Se declară figuri şi tabele (cu stiluri stabilite anterior) folosind *captură* („Insert – Reference - Caption”), pentru a iniţia numerotarea automată a acestora.

vi) Se fac *trimiteri* în text la figuri şi Tabele *prin legarea* acestora de captură („Insert – Reference – Cross-reference”). Trimiterile se pot face pentru orice fel de elemente declarate anterior prin captură. În acest mod, orice modificare în ordinea elementelor capturate nu afectează şi legăturile cu ele în text (adică dacă s-a mai introdus o figură numerotată înaintea uneia legate, se actualizează şi trimiterea din text la figura legată).

vii) Se activează verificarea ortografică şi gramaticală.

viii) Se scrie conţinutul documentului fără a da mare atenţie formei, pentru a se focaliza atenţia pe idei.

ix) Se verifică corectitudinea textului (urmărind sublinierile şerpuite v. §5.1.4) şi formatarea conform stilurilor prestabilite.

5.2 Foi de calcul tabelar

Foile de calcul tabelar („spreadsheet”) sunt poate cele mai puternice instrumente de calcul la dispoziţia utilizatorilor obişnuiţi. De la crearea lor – de către Dan Bricklin şi Bob Frankston prin produsul VisiCalc, au apărut o mulţime de alte produse (Lotus 1-2-3, Quattro Pro, Excel) care au preluat aproape neschimbate ideile creatorilor, extinzând facilităţile foii de calcul – cu funcţii, prelucrări analitice (de detaliere) şi sintetice (de modelare şi sinteză).

O *foaie de calcul tabelar* (numită şi *foaie electronică de calcul*) este o imitare a unui registru contabil prevăzut cu caroiaj pentru a înşira coloane de cifre şi linii cu semnificaţii de valori de obiecte sau operaţiuni, însă forma electronică permite înscrierea directă în foaie a unor formule care sunt „vii” – adică îşi modifică valoarea imediat ce unul din operanzi s-a modificat. Mai mult, se pot realiza şi afişa uşor grafice complexe, tabele similare celor din baze de date, programe de prelucrare locală sau de automatizare a unor operaţiuni folosind macrocomenzi.

5.2.1 Structura foii de calcul tabelar

Fereastra de calcul tabelar – vezi figura demai jos, prezintă un set de *foi de calcul* (14) („Sheets”) grupate într-un *caiet de calcul* (1) („Workbook”). Fiecare foaie de calcul are un nume (implicit denumite Sheet1, Sheet2, ...) între care una este activă (13)(Sheet1). Foaia de calcul este împărţită prin caroiaj pe *rânduri* numerotate (12) şi coloane denumite cu litere (4), ce delimitează *celule* (15) („Cells”).

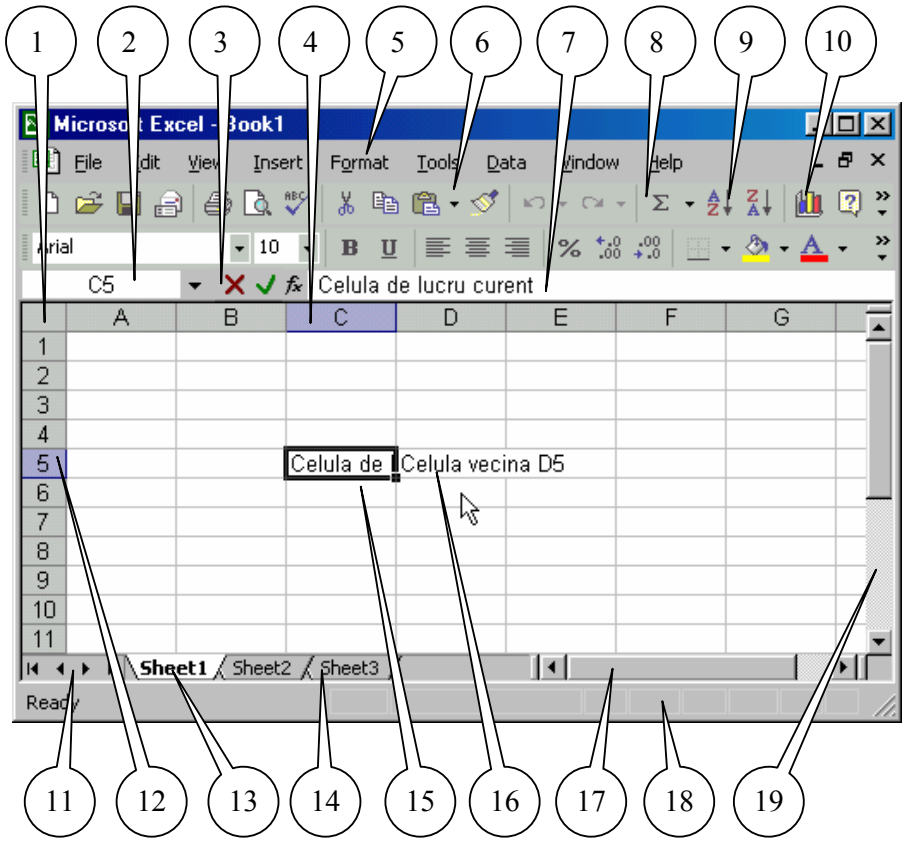
După cum se observă în figură, într-o celulă se pot înscrie informații – de exemplu în (15)șirul „Celula de lucru curent” care însă nu este vizibil integral fiind acoperit de (capul) șirului din celula (16). Fiecare celulă prezintă o adresă (referință) specificată prin denumirea coloanei și numărul rândului – de exemplu celula curentă (15) are adresa **C5**. Celulele pot fi grupate în zone rectangulare denumite *blocuri* de celule (sau domenii - „ranges”); un bloc este referit prin adresele celulelor din colțurile stânga-sus și dreapta jos – de exemplul blocul celulelor vizibile în fereastra este referit **A1: G11**.

În structura unui *caiet de calcul* se pot afla: *foi de calcul* („work sheet”) – ce conțin celule cu informații, *foi de grafice* („chart sheet”) – ce conțin grafice denumite, *foi cu macro-comenzi* („macro sheets”) – ce conțin comenzi în limbajul specific foilor de calcul. Derularea foilor din cadrul caietului de foi se face prin butoanele (11), foaia curentă (13)fiind denumită *foaie activă*.

Elementele din structura ferestrei caietului de calcul sunt:

(5) *Bara de meniu* specifică, (6) *Bara de unelte* cu butoane specifice pentru funcții (8) (sumare), sortare (9), sau grafice (10), apoi *bara de stare* (18) și *bare de derulare* (17) și (19)- similare altor aplicații.

Elementele din structura ferestrei foii de calcul sunt:



(2) *Bara de nume* – ce indică numele celulei sau blocului de celule sau, în cazul lipsei numelui, adresa celulei curente;

(7) *Bara de formule* – ce indică conținutul celulei curente și permite scrierea unei formule; după introducerea completă a formulei, în celula curentă din foaie se afișează rezultatul formulei iar în bara de formule expresia acesteia.

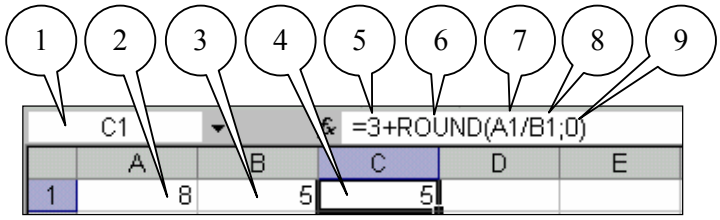
③ *Bara de editare* – ce permite editarea conținutului celulei curente introducând, de exemplu, o funcție dorită în formulă prin pe , confirmarea editării prin pe sau renunțarea la editarea celulei prin pe („Escape”).

Foaia de calcul prezintă un număr de 256 coloane, denumite cu literele alfabetului, apoi cu litere duble: A, B, C,…Z, AA, AB, AC,…AZ, BA, BB, BC,…BZ, CA,…IA, IB,…IV şi 65.536 de rânduri, cu celule suficiente pentru volume mari de date.

5.2.2 Conținutul celulelor foii de calcul

În celulele foii de calcul se pot înscrie date de diferite tipuri, dar şi prelucrări, care sunt afişate conform formatării celulei (opțiunea „Format - Format Cell”). Atât datele cât şi prelucrările sunt „recunoscute” şi tratate corespunzător:

- *tip General* – celula nu are un format definit de număr.
- *Număr* – conține numai cifre şi este aliniat la dreapta implicit. Afişarea unui număr mare se poate face prin mantisă şi exponent (v. §2.3.2.1.) – dacă prin lungimea sa nu încape (deci nu este vizibil integral) în celulă; de exemplu numărul 15.000.000.000 va apare 1.50E+10. Se poate indica numărul de zecimale şi modalitatea de afişare a numărului: număr obişnuit („Number”), însoțit cu simbol de monedă („Currency”), procent („Percentage”), în format ştiințific („Scientific”), ca număr special („Special” de exemplu cod poştal, număr de telefon, cod numeric personal).
- *Şir de caractere* („Text) – conține orice simboluri de scriere şi începe cu o literă, fiind aliniat la dreapta implicit. Într-o celulă se poate înscrie un şir de maxim 256 caractere.
- *Dată şi oră* – reprezentate ca numere astfel: data ca întreg indicând numărul zilei începând cu 31 Dec. 1899, iar ora ca număr zecimal fracție a zilei (de 24 ore) – de exemplu ora 12 la prânz reprezentată ca 0,5. Formatul de afişare a datei şi orei este însă ales după dorință d utilizator: ex. 15.03.2003, 15 martie 2003, 15.03.03, etc.
- *Formulă* – expresie care începe cu semnul ‘=’, conține operanzi, operatori şi funcții. Operanzii pot fi literali (valori directe) sau adrese ale celulelor cu care se doreşte efectuarea calculelor.



Exemplu de structură a formulei.

Pe exemplul demai sus, se prezintă cele mai importante caracteristici ale unei formule şi modul cum se prezintă ea în foaia de calcul:

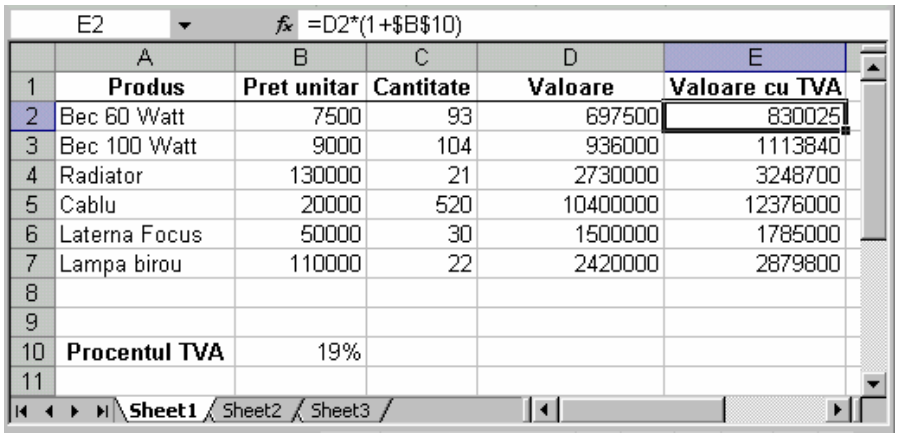
- Formula este înscrisă în celula C1 (vezi ④), cu adresa indicată de caseta de nume ①
- Conținutul celulei C1 este vizibil în bara de formule iar în foaia de calul (celula C1) apare valoarea rezultat a formulei.
- Operanzii formulei sunt literalul ⑤ şi funcția ⑥ (ROUND - rotunjire) care are doi parametri: primul parametru este valoarea de rotunjit raportul între celulele A1 şi B1 (vezi ⑦ şi ⑧) iar al doilea ⑨ este numărul de zecimale la care se face rotunjirea – în exemplul dat 0 adică rotunjire la întreg.

5.2.3 Lucrul cu foaia de calcul

În general, foile de calcul sunt utile pentru prelucrarea unor blocuri de valori, care se înscriu în celule şi sunt folosite ca date de intrare pentru expresii (formule) aflate în alte celule. Blocurile cu rezultate pot fi în final reprezentate grafic (prin histograme).

5.2.3.1 Adrese relative şi absolute

Datele de tipurile prezentate mai sus şi formulele se înscriu în celule şi se pot edita prin manevre similare celor prezentate la procesoare de texte, adică prin marcare urmată de copiere sau mutare. În plus, în foaia de calcul, se pot copia o celulă şi multiplica într-un bloc de celule. (*copii multiple*); această facilitate este foarte utilă la multiplicarea formulelor care se aplică similar mai multor seturi de date.



	A	B	C	D	E
1	**Produs**	**Pret unitar**	**Cantitate**	**Valoare**	**Valoare cu TVA**
2	Bec 60 Watt	7500	93	697500	830025
3	Bec 100 Watt	9000	104	936000	1113840
4	Radiator	130000	21	2730000	3248700
5	Cablu	20000	520	10400000	12376000
6	Laterna Focus	50000	30	1500000	1785000
7	Lampa birou	110000	22	2420000	2879800
8					
9					
10	**Procentul TVA**	19%			
11					

Exemplu de multiplicare şi utilizare a adreselor relative sau absolute.

În exemplul de mai sus se prezintă un tabel de produse pentru care se calculează valoarea fiecăruia. Formula înscrisă în coloana „Valoare" celula D2 este un produs între „Preţ unitar" şi „Cantitate" la fiecare produs; formula de calcul se scrie o singură dată în D2: = B2*C2, iar apoi se replică în celulele D3:D7 prin *copiere multiplă*. La această operaţie de copiere, formulele se actualizează automat (privind adresele operanzilor) relativ la noua poziţie pe care o are formula. Astfel, în celula D3 se obţine (prin copiere) = B3*C3, în D4 = B4*C4 şi aşa mai departe. Această modalitate de copiere este evident avantajoasă în cazul formulelor şi este posibilă fiindcă în mod special adresarea unei celule prin indicarea simplă a liniei şi coloanei este o *adresare relativă*.

Există şi situaţii în care adresarea relativă nu este avantajoasă, de exemplu calculul valorii cu TVA indicată în figură. Celula B10 conţine procentul de TVA şi apare ca operand în E2, cu denumirea coloanei şi numărul liniei precedate de simbolul special $, care are ca efect transformarea adresei B10 în *adresă absolută*. Astfel, operandul indicat prin B10 nu va fi actualizat la copierea în alte celule, adică va rămâne indicat în mod absolut. În acest fel se poate evita înscrierea procentului TVA în mod explicit în fiecare din formule – adică în E2 ar fi =D2*(1+0.19) – valoarea TVA înscrisă ca literal, dar nici nu ar trebui înscrisă pentru fiecare din rânduri (spre a folosi celule operand cu procent TVA – dedicat fiecăruia din rânduri). Mai mult, dacă valoarea procentului TVA s-ar modifica, nu este nevoie să se intervină în fiecare din celulele care o folosesc (cum ar fi cazul pentru cele două exemple descrise mai sus) ci s-ar modifica o singură dată celula B10 – cu noua valoare a procentului.

5.2.3.2 Funcţii

Produsele foi de calcul electronic conţin multe funcţii – fiecare dedicată unei prelucrări anume, care poate fi apelată într-o formulă (ca un subprogram) prin *nume* şi *parametri* între

paranteze, furnizând rezultatul corespunzător. Parametrii pot fi: literali, adrese de celule (de ex. D2) şi/sau adrese de blocuri (de ex. D2:D7) – ca în exemplul de mai sus.

Categoria	Exemple de funcţii
Matematice	ABS(*numar*) – furnizează valoarea absolută pentru *numar*. SIN(*numar*) – furnizează valoarea sinusului pentru *numar*. ROUND(*numar*, *nr_zec*) – furnizează rontunjirea *numar* la *nr_zec* zecimale. EXP(*numar*) – furnizează *e* (Neper) la puterea *numar*. COMBIN(*n,m*) – furnizează combinaţii de *n* luate câte *m* (C^n_m).
Logice	AND(l_1, l_2, ...) – furnizează valoarea logică TRUE dacă toate valorile logice l_1, l_2, ... (maxim l_{30}) sunt TRUE – v. §2.3.4 OR(l_1, l_2, ...) – furnizează valoarea logică FALSE dacă toate valorile logice l_1, l_2, ... (maxim l_{30}) sunt FALSE – v. §2.3.4 IF(*expr_logica*, *val_DA*, *val_NU*) – evaluează expresia logică *expr_logica* şi la rezultat TRUE furnizează *val_DA* altfel *val_NU*.
Statistice	AVERAGE(n_1, n_2, ...) – furnizează media aritmetică a numerelor n_1, n_2, ... (maxim n_{30}). VAR(n_1, n_2, ...) – furnizează abaterea medie pătratică (varianţa) pentru numerele n_1, n_2, ... (maxim n_{30}). MAX(n_1, n_2, ...) şi MIN(n_1, n_2, ...) – furnizează maximul respectiv minimul numerelor n_1, n_2, ... (maxim n_{30}).
Financiare	AMORDEGRC(cost, data_ac, per1, val_r, per, rata, baza) – furnizează amortizarea pentru numărul *per* de perioade, conform ratei de amortizare *rata* pe o perioadă (lună, an), pentru bunul achiziţionat la *data_ac* şi sfârşitul primei perioade de amortizare *per1*, cu baza de calcul *baza* (o valoare între 0 şi 4 pentru 360 până la 365 zile). IRR(*valori*, *eval*) – furnizează rata internă de revenire a unui flux de capital (sumele predate la intervale fixe – lună, an), adică dobânda pentru o investiţie cu depuneri periodice (sume negative) sau un venit periodic (sume pozitive); *eval* este estimarea dobânzii dorite. FV(*dob*, *nper*, *prima*, *pv*, *mom*) – furnizează valoarea viitoare a unei investiţii cu dobânda *dob* făcută prin plăţi periodice *prima*, pe un număr *nper* perioade (pentru fiecare primind dobânda *dob*). Parametrii opţionali *pv* şi *mom* indică valoarea evaluată în prezent a unei serii de investiţii şi momentul când se face plata primei (începutul sau sfârşitul perioadei). PMT(*dob*, *nper*, *pv*, *fv*, *mom*) – furnizează valoarea ratei (ca sumă plătită) pentru un împrumut cu plăţi periodice (v. parametrii mai sus). NPER(*dob*, *pmt*, *pv*, *fv*, *mom*) – furnizează numărul de perioade pentru o investiţie cu plăţi *pmt* şi dobândă *dob* constante (v. parametrii mai sus). IPMT(*dob*, *per*, *nper*, *pv*, *fv*, *mom*) – furnizează valoarea dobânzii plătite pentru o ipotecă cu plăţi periodice în perioada *per* (număr între 1 şi *nper* – numărul total de perioade), pentru un împrumut *pv (fv* şi *mom* sunt opţionali, cu semnificaţii de mai sus).
Baze de Date (Tabele)	Toate funcţiile au ca argumente: blocul ce conţine *baza* de date, *coloana* (câmpul) prelucrat şi blocul unde se află *criterii*le de selecţie. DAVERAGE(*baza*, *coloana*, *criteriile*) – furnizează media valorilor din *baza* de date pentru *coloana* şi *criteriile* specificate. DSUM(*baza*, *coloana*, *criteriile*) – furnizează suma valorilor din *baza* de date pentru *coloana* şi *criteriile* specificate. DMAX(*baza*, *coloana*, *criteriile*) – furnizează maximul valorilor din *baza* de date pentru *coloana* şi *criteriile* specificate.

În tebelul de mai sus se prezintă categorii de funcții comune accesibile în foi de calcul electronic, cu câteva exemple de funcții în fiecare categorie.

Funcțiile se utilizează în formule care, în general, sunt multiplicate în fiecare celulă din coloana care se doreşte să conțină rezultatele, iar datele de prelucrare se găsesc în coloane alăturate.

5.2.4 Facilități de prezentare calitativă şi sintetică a informațiilor

Pentru a manipula datele dintr-o foaie de calcul electronică şi pentru a extrage informații utile deciziilor, acestea trebuie prelucrate şi prezentate în mod sintetic şi calitativ, pentru utilizatorul să aică o imagine intuitivă. Unele dintre acestea se prezintă succint în continuare.

5.2.4.1 Grafice

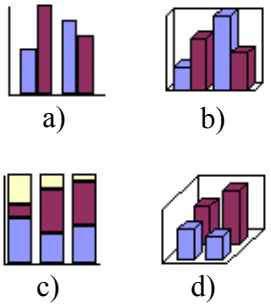
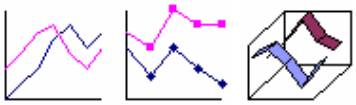
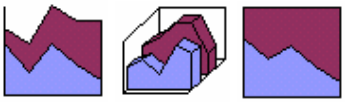
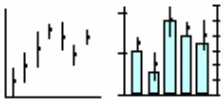
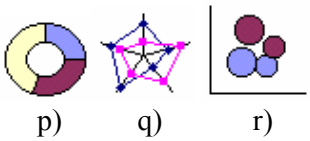
O altă facilitate forate utilă în foile de calcul electronic o reprezintă graficele. Acestea sunt folosite pentru a prezenta în mod calitativ şi sintetic informația numerică, sub diferite forme, astfel că se pot compara „dintr-o privire" situații cantitative. Graficele sunt o reprezentare a uneia sau mai multor serii cantitative, relative la momente sau situații discrete date. O serie cantitativă este o listă de numere (înscrisă pe o linie sau pe o coloană) în care elementele indică starea la un moment dat în *evoluția unei mărimi* (de exemplu vânzări lunare ale unui produs pe fiecare lună din an) sau *valorile statice ale mai multor mărimi* (de exemplu proporția – în procente, a vânzărilor primelor 5 cele mai vândute produse). Privind reprezentarea grafică a numerelor, se pot face uşor şi clar o comparații sau se pot stabili relații calitative între mărimi cantitative – necesare în luarea deciziilor.

Reprezentarea grafică a seriilor de numere se poate face în diferite sisteme de coordonate:

- *Carteziene* – sistem de axe perpendiculare în două sau trei dimensiuni (2D sau 3D). De obicei axa orizontală transversală (pe lățime) este denumită „Axa X" reprezintă situațiile de referință (de ex. momente de timp în evoluția mărimii ilustrate grafic). Axa verticală – „AxaY", este cea pe care se reprezintă cantitățile (numerele efective din serie). La graficele 3D există o a treia axă – „Axa Z", orizontală în profunzime, care indică fiecare serie atunci când se ilustrează mai multe serii (mai multe mărimi) simultan pe acelaşi grafic.
- *Polare* – sistem de reprezentare circulară în două dimensiuni, care ilustrează intuitiv cantități comparative ce împreună fac un întreg (ca de exemplu grafic „tort" în care fiecare felie reprezintă vânzarea unui produs pe o lună anume din întregul an). Graficele polare au efecte 3D dar numai pentru o ilustrare mai estetică.

Graficele pot fi realizate rapid şi apoi se pot completa cu toate informațiile utile folosind un asistent („wizard") care conduce utilizatorul prin paşi, fiecare prezentând o casetă de dialog:

a. Alegerea tipului de grafic – dintr-o listă de tipuri; în tabelul de mai jos se prezintă aceste tipuri şi situațiile în care sunt utile fiecare.
b. Indicarea seriilor de numere ca blocuri în foaia de calcul, pentru axele X, Y şi Z. La graficele cu mai multe serii în acelaşi sistem de coordonate se denumeşte fiecare în parte.
c. Completarea graficului cu informații suplimentare lămuritoare şi plasarea acestora în imaginea finală: titlul şi subtitlul graficului, denumirea axelor şi marcarea diviziunilor pentru mărimile cantitative pe care le reprezintă, legenda, alte informații lămuritoare (valorile numerice efective, etichete cu comentarii ataşate unor valori din serie), culori şi modele grafice de haşură (linii paralele trasate oblic, orizontal, vertical) sau poşare (umbrire sau culoare uniformă sau în degradé).
d. Denumirea şi plasarea graficului în foaia de calcul sau într-o foaie separată, dedicată (foaie de grafic).

Denumire grafic, cu caracteristici şi situaţii de utilizare	Reprezentare grafică
Grafic de bare: verticale („Columns”) sau orizontale („Bars”), în coordonate carteziene: a) pentru o singură serie, 2D b) pentru o singură serie, cu efecte estetice 3D; c) pentru serii multiple 2D, cu coloane suprapuse; d) pentru serii multiple 3D, cu seriile pe axa Z. Utilizate pentru a ilustra *evoluţii în timp discret* (de ex. volumul vânzărilor pe luni) sau *situaţii discrete* (de ex. vârstele persoanelor dintr-un grup).	 a) b) c) d)
Grafic linie („Line”): linii în coordonate carteziene: e) linii pentru una sau mai multe serii 2D, f) linii cu punte marcate pentru serii 2D, g) pamblică pentru serii 3D. Utilizate pentru a ilustra *evoluţii continue* în timp.	 e) f) g)
Grafice plăcintă („Pie”) – ca un tort cu felii, în coordonate polare pe un cerc închis: cu felii „explodate” h) plăcintă 2D compactă; i) plăcintă 3D pentru efect estetic. j) plăcintă cu „felii explodate”; Utilizate pentru ilustrarea de *mărimi statice ce fac un întreg* (de ex. vânzările pe trimestre ale unui an).	 h) i) j)
Grafice arie („Area”): k) arie 2D; l) arie 3D m) arie comparativă: o mărime reprezintă 100% iar cealaltă proporţia din ea. Utilizate pentru ilustrarea unor mărimi cu caracter integrator sau cumulativ (de ex. pentru evoluţia investiţiilor).	 k) l) m)
Grafice curs („Stock”)– ca segmente sau bare cu liniuţe: n) curs maxim-minim-la închidere („high-low-close”) care indică valorile maxim şi minim prin capetele de segment verticale, iar valoarea la închidere prin liniuţa laterală; o) curs volum-maxim-minim-la închidere („volume-high-low-close”) ca mai sus, în plus volumul tranzacţiilor indicat prin înălţimea barei verticale. Utilizate pentru ilustrarea evoluţiei zilnice a acţiunilor sau tranzacţiilor financiare (eventual şi volumului).	 n) o)
Grafice speciale – care sunt variante ale celor amintite până aici dar cu alte forme decât cele „clasice”: p) gogoaşă americană („Doughnut”) – grafic în coordonate polare, similar celui plăcintă; q) radar („Radar”) – grafic polar prin segmente în locul sectoarelor de cerc ale plăcintei, pe mai multe axe; r) Buline („Bubble”) – grafic polar şi cartezian prin buline în locul sectoarelor de cerc ale plăcintei (pentru cantităţile seriei) şi mărimi de referinţă pentru de seria de bază pe axele carteziene.	 p) q) r)

5.2.5 Indicații de proiectare şi construire a foilor de calcul

După cum s-a arătat până aici, foile de calcul electronic oferă mijloace puternice şi uşor de folosit pentru prelucrări complexe. De exemplu, se poate analiza starea şi perspectiva unei afaceri (folosind şiruri de date legate privitoare la situațiile financiare sau tehnice ale firmei, cu formule şi funcții adecvate), se poate face o analiză a variantelor de dezvoltare a afacerii şi, în final, se poate prezenta calitativ atât starea actuală cât perspectiva prin grafice intuitive şi printr-un document tipărit, generate cu ajutorul produsului foaie de calcul.

Totuşi, pentru a structura corect şi a prelucra eficient un volum mare de informații, este necesar un mod de lucru sistematic iar apoi utilizarea corectă şi completă a mijloacelor puse la dispoziție de foaia de calcul; în acest scop se prezintă succint recomandări în acest subcapitol. Trebuie remarcat din capul locului, că foile de calcul sunt utile la rezolvarea problemelor ce implică date diverse, statistici şi calcule, cu ilustrări calitative (grafice) şi sintetice (sume de total, tabele pivot, etc.). Paşii de lucru urmați în rezolvarea problemei cu foaia de calcul sunt cei indicați la §4.1.1 – proprii realizării unei aplicații.

5.2.5.1 Formularea cerințelor şi analiza problemei

Cunoscând *problema de rezolvat şi scopul acesteia*, se face un plan corect, care se bazează pe datele (eventual statisticile) de care se dispune în prezent şi cele necesare în rezolvarea problemei. Se face un *inventar al funcțiilor* şi ce vor fi utilizate în prelucrarea datelor şi în compararea rezultatelor, apoi se stabileşte cum se doreşte *prezentarea rezultatelor* (grafic, tabele, rapoarte). Se analizează şi se stabilesc:

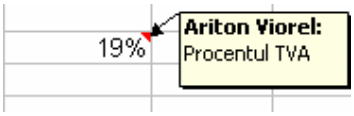
- Intrările: *valori cunoscute* – cele ce urmează a fi prelucrate efectiv în foaie, *variabile de decizie* – cele prin care se pot alege variante şi se poate optimiza modelul (de exemplu utilizând scenarii.
- Ieşirile: *variabile obiectiv* – care se doresc calculate (prin formule, funcții) şi *variabile de restricție* – care impun limite sau condiții de compromis ce trebuie respectate (de exemplu limite de buget).
- Relațiile: *relații între variabile* cunoscute cele de decizie (care le condiționează pe primele) şi utilizarea unui *şablon* – oferit de produsul foaie de calcul sau preluat din alte aplicații (de exemplu problemele de alocare a resurselor au un format tradițional).

Aceste date sunt preluate de pe documente scrise sau din fişiere aflate pe disc, în ultimul caz foile de calcul electronic oferind mijloace de import (din baze de date sau din fişiere text – de exemplu de tip CSV adică date înscrise pe linii şi separate prin spații, şirurile de caractere între ghilimele). Relațiile se identifică şi se înscriu pe hârtie, în documentul de analiză. La realizarea foii de calcul sunt necesare, uneori, variante de lucru şi calcule suplimentare; acestea este bine să se facă în altă foaie de calcul – foaie de manevră, care va fi salvată sau nu alături de foaia (caietul) de bază.

5.2.5.2 Proiectarea şi realizarea foii de calcul

Date fiind mijloacele puse la dispoziție de o foaie de calcul electronică, cele două etape: proiectarea şi realizarea aplicației, se parcurg simultan; în plus, este indicată documentarea foii de calcul pe măsura realizării ei – aşa cum se prezintă în paşii ce urmează:

1. **Introducerea şi organizarea datelor**. Se introduc datele cunoscute chiar dacă încă nu se ştie exact cum vor fi folosite. Astfel, se înscriu: *valori constante* – care este bine să fie păstrate într-o zonă separată a foii, denumite corespunzător.
2. **Documentarea foii**. Este bine ca în cadrul foii să se înscrie textul de descrie problema, scopul şi modalitatea rezolvării ei, ca *documentare generală*. În plus, celulelor importante li se pot adăuga nume şi *comentarii explicite* (prin meniul „Insert-

Comment"), aceste celule fiind indicate cu un colț roşu 

şi prezintă o etichetă vizibilă la indicare cu mouse .

3. **Crearea formulelor**. Înscrieți formule ori de câte ori este nevoie; acestea se pot copia multiplu şi edita specific locului unde apar iar funcțiile oferite acoperă un spectru larg de utilizări. *Evitați înscrierea numerelor* (literali) direct în formule, mai bine se fac referiri la celule care conțin valorile respective (exemplu la final §5.2.3.1 cu valoarea TVA). Se recomandă *folosirea numelor în loc de adrese de celule* (pentru ca formula să fie uşor inteligibilă) şi se păstrează rezultate intermediare (de ex. subotaluri); în general este bine ca *formulele să fie simple* – folosind rezultate intermediare mai degrabă decât a înscrie o formulă complicată, ne-inteligibilă.
4. **Prezentarea rezultatelor**. Datele obținute prin calcule nu sunt uşor de urmărit şi înțeles dacă acestea nu se prezintă şi în *forme sintetice* (totaluri sau chiar modele matematice) şi *forme calitative* (grafice). În final, documentul cu tabele şi imagini se poate tipări, adăugând *antet şi subsol, titluri, formatare text şi liniaturi la tabele*.

Pentru obținerea rezultatului urmărit se poate apela la mijloace de analiză şi sinteză pe loc – prin tabele pivot, regresii sau analize statistice, care ajută la găsirea metodei de soluționare a problemei.

5.3 Rezumat

Cea mai uzuală utilizare a calculatorului este cea pentru aplicații de birou, care cuprind: crearea de documente, structurate şi scrise în echipă (folosind procesoare de texte), calcule diverse şi situații tabelare ce conțin valori şi calcule (folosind foi de calcul tabelar), prezentări către auditoriu şi comunicații. Primele două categorii de aplicații sunt prezentate în acest capitol, în care se indică nu doar funcțiile oferite de acestea ci şi modul de utilizare a lor. Procesoarele de texte permit editarea şi formatarea paragrafelor şi caracterelor, stabilirea structurii de titluri, trimiteri şi etichete (pentru figuri şi tabele) din document, inserarea de imagini şi desene simple. Elemente uzuale în text care necesită numerotare (cum sunt listele, titlurile, paginile sau etichetele de figuri şi tabele) sunt gestionate automat de procesorul de texte dacă aceste elemente s-au declarat cu mijlocele puse la dispoziție de procesor. Tabele, liniaturi şi alte structurări ale textului (precum adânciri de paragrafe, spații, etc.) pot fi realizate automat şi utilizate în acelaşi fel în alte produse de tip procesor de texte. Foile de calcul electronic permit structurarea informațiilor în celule în formă tabelară, după care se pot crea formule ce au ca operanzi adrese ale unor celule, literali şi funcții. Seriile de date numerice pot fi ilustrate calitativ prin histograme (grafice) de diverse tipuri, care se realizează simplu şi eficient, în plus putând fi transferate în alte documente (de exemplu scrise cu procesorul de texte). Se prezintă detalii despre modul cum se operează cu foi de calcul tabelar.

5.4 Teme de control

1. Dați exemple de elemente din structura de conținut a unui document scris.
2. Dați trei exemple de elemente de text ce necesită numerotare, care pot fi automatizate cu ajutorul procesorului de texte?
3. Dați trei exemple de indexe şi tabele („Index and Tables ...") ce pot fi realizate automat de un procesor de texte.
4. Ce înseamnă indentarea paragrafelor? Ce mijloace de indentare are procesorul de texte?
5. Dați un exemplu de listă numerotată şi indicați modul cum poate fi creată.

6. Cum se activează verificarea ortografică şi gramaticală pentru un document?
7. Se poate face corectură automată a textului pentru erori sistematice? Argumentaţi.
8. Ce este o celulă şi ce este un bloc de celule într-un produs de calcul tabelar?
9. Daţi exemple de operanzi „adresă de celulă" şi literali pe un exemplu de formulă în Excel.
10. Ce au în comun şi ce au diferit specificarea adreselor relative şi absolute?
11. Daţi exemplu de o funcţie matematică şi una financiară cu rolul a doi parametri la fiecare.
12. Când este indicată folosirea graficelor tip coloană şi când a celor tip tort?
13. Când este indicată folosirea graficelor cu linii continue şi când folosirea graficelor de tip bară?

Exemple de răspuns:

(2) Exemple de elemente de text ce necesită numerotare: liste numerotate, numărul de pagină, numărul (ca idice superior) al unei note de subsol. Aceste elemente se numerotează automat dacă sunt realizate cu ajutorul istrumentelor dispoibile în procesorul de texte: Liste – meniu Format-Buletts and Numbering; Număr de pagină – meniu Insert-Page number; Note de subsol – meniu Isert-Reference-Footnote.

(7) Da, se poate face corectura automată a unor cuvinte greşite în acelşi mod (sistematic), folosind opţiunea Tools-AutoCorrect Options, unde se înscrie forma eronată în coloana „Replace" iar forma corectă în coloana „With".

6 Rețele de calculatoare şi comunicații

Organizațiile actuale – de producție şi comerț, administrative, de educație sau cu scop caritabil, manipulează informații care circulă între membrii acestora, între departamente sau sedii proprii, schimbă informații cu alte organizații. Aceste informații trebuie să fie transferate la modul sigur, în timp scurt şi să permită totodată comunicare interactivă între entități, fie acestea oameni sau aplicații pe calculator.

Comunicațiile pot avea loc prin voce (de exemplu prin telefon), imagine (video), pe suport hârtie (documente şi texte în diferite formate) sau prin date reprezentate numeric pentru prelucrarea directă pe calculator. Tendința actuală este de integrare a acestor categorii de comunicații prin digitizarea informației şi transferul acesteia prin medii diferite (cabluri de cupru, fibre optice, eter), sub formă de unde electromagnetice. Se pot transmite prin aceleaşi medii de transfer atât date numerice cât şi sunete sau imagini realizând astfel *comunicarea totală* între entități .

6.1 Problematica rețelelor de calculatoare

Studii de trafic telefonic arată că 80% din comunicații au loc în interiorul întreprinderii şi 20% în afara sa. Pe această constatare se bazează şi adoptarea tehnicilor implicate în comunicațiile de date: pentru a asigura traficul intens în interiorul întreprinderii se folosesc aşa numite rețele locale - bazate pe medii şi tehnici speciale şi cu rată mare de transfer, iar pentru traficul spre exterior se folosesc rețele largi - bazate pe infrastructura unor operatori (firme de telecomunicații), care prezintă rată mai mică de transfer dar la distanțe mari. Aplicații multimedia sau de educație la distanță necesită „bandă largă" (debit mare de date) astfel că astăzi se extind tehnologii de transfer de mare viteză – pe fibră optică sau cupru.

Un calculator poate fi singular („stand-alone") sau stație de lucru conectată în rețea („workstation"); diferențele între acestea constau în piesele hardware şi software adăugate primului pentru a obține pe al doilea. Pentru PC-uri aceste piese sunt în general o cartelă electronică (placă de rețea sau modem) şi pachete software "client", adică programe care rulează pe maşina utilizatorului şi oferă acestuia accesul la servicii prin comunicații în rețea. Între servicii se pot enumera: oferirea spațiului de stocare pe discuri, prelucrarea la distanță pe maşini puternice pentru calcul intensiv, tipărire la distanță, acces la informații aflate pe maşini distante, comerț electronic, etc.

Schimburile de date şi prelucrarea pe mai multe procesoare comportă două abordări principale – privind modalitatea de interconectare a elementelor de prelucrare:

- Interconectare prin *cuplaj strâns*, în care procesoarele împart o magistrală sau o memorie comună, rata de transfer a datelor este mare (cea a magistralei interne a procesorului), iar distanțele sunt foarte mici (centimetri). Elementele de prelucrare se află în general în aceeaşi carcasă.
- Interconectare prin *cuplaj slab*, în care procesoare independente schimbă date prin interfețe specializate (interfețe de comunicație) şi prin medii de transfer exterioare, pe distanțe la nivelul clădirilor sau la nivel global.

În general, se înțelege prin *rețea de calculatoare* un ansamblu de elemente de prelucrare cuplate slab. Elementele de prelucrare pot fi calculatoare (cu putere de calcul mai mare sau mai mică) dar şi imprimante, arii de discuri, instalații pentru comunicație prin cablu coaxial, fibră optică sau instalații pentru comunicații prin satelit.

6.1.1 Componente ale rețelelor de calculatoare

Transferul datelor în rețele de calculatoare se face doar prin unde electromagnetice, în spectrul invizibil (unde radio, unde infraroşu şi microunde) sau în cel vizibil (lumină).

În structura rețelei de calculatoare intră diverse echipamente şi suportul undelor electromagnetice (formând infrastructura de comunicație) În cele ce urmează se face un inventar scurt al componentelor rețelelor de calculatoare:

6.1.1.1 Echipamente de prelucrare a datelor

Aceste echipamente găzduiesc programele de prelucrare sau efectuează servicii diverse către utilizatorul uman şi pot fi:

- *stație de lucru* („work station") – calculator la care lucrează utilizatorul uman;
- *server* – calculator care oferă diverse servicii utilizatorilor în rețea;
- *imprimantă de rețea* – care oferă pe lângă serviciul de imprimare şi gestionarea lucrărilor de tipărire;

Modul de lucru uzual în rețele de calculatoare este bazat pe *arhitectura client-server*, în care programul de lucru este împărțit în două părți (client şi server) ce conlucrează între ele (v. §4.2.2.3). Există însă şi modalități de lucru în *arhitectura gazdă-terminal* („host-terminal"), în care întreg programul se execută pe o maşina gazdă (care rulează logica aplicației şi găzduieşte datele) iar operațiile de intrare / ieşire spre utilizatorul uman se execută de terminal.

6.1.1.2 Mediu de transfer

Undele electromagnetice transportă date între echipamentele în rețea dar ele necesită un suport de propagare care poate fi:

- *cablu* – pentru propagarea undei electromagnetice în spectrul invizibil (prin cupru) sau în spectrul vizibil prin (fibră optică);
- *eter*[1] – spațiul deschis atmosferic sau cosmic (vid), asigurând comunicații în spectrul invizibil (radio, infraroşu, microunde) sau în spectrul vizibil (laser).

În oricare din situații, o secțiune de mediu neîntreruptă se numeşte *segment*; acesta poate fi segment de cablu sau doar spațiul dintr-o încăpere prin care se face legătura directă între stații interconectate prin infraroşu, de exemplu.

Cablurile de cupru sunt de diferite tipuri, dependent de numărul de fire şi modul lor de plasare în cablu (de exemplu prin împletire – fire torsadate), de gradul de ecranare față de perturbații din exterior, de tipul mantalei de izolație. Tipul de cablu folosit la realizarea unei rețele se alege după cerințele de performanță ale rețelei în compromis cu costul cablului.

6.1.1.3 Echipamente de interconectare

Pentru direcționarea pachetelor de date pe drumul către destinație, pe unul din mediile de propagare din rețea, se folosesc echipamente care au rol de dispecer – denumite echipamente de interconectare. Acestea sunt adesea construite în jurul unui sistem de calcul dedicat, pe care nu lucrează utilizatorul uman (eventual se poate conecta administratorul de rețea doar pentru a realiza anumite configurări), între carese amintesc:

- *repetor* („hub") – care practic prelungeşte două sau mai multe segmente ale mediul de transfer, realizând amplificare de semnal şi intervenții sumare în pachetele de date;
- *comutator* („switch") – care recepționează, memorează şi înaintează pachetele de date spre segmente conectate direct la alte echipamente (de prelucrare sau interconectare);

[1] noțiune apărută în secolul XIX indicând un suport material pentru propagarea undelor în vid, care însă nu a fost susținut de experimente

- ♦ *dirijor* („router”, numit impropriu „gateway” în jargonul Internet) – care interconectează rețeaua locală (LAN) la rețeaua largă (WAN), de obicei printr-un operator de servicii de comunicații.

Echipamentele de interconectare asigură canale de transfer a datelor între echipamentele de prelucrare şi în plus pot mixa diverse medii de propagare (cabluri de cupru, fibră optică, eter). Astăzi, aproape toate tipurile de echipamente de interconectare asigură şi posibilități de administrare a resurselor în rețea, adică: configurarea de la distanță a diverselor echipamente, auto-anunțarea defectului, locului şi momentului în care a survenit, identificarea tipului de echipament şi producătorului pentru generarea automată a schemei actuale de configurare a rețelei. Aceste mijloace şi software-ul adecvat sunt instrumente puternice în mâna administratorului de rețea, pentru asigurarea funcționării corecte şi neîntrerupte a rețelei.

6.1.2 Clasificarea rețelelor de calculatoare după extindere spațială

Extinderea spațială a rețelei este strâns legată de utilizarea sa; rețeaua unei organizații sau de departament este restrânsă spațial, dar prezintă şi caracteristici de acces şi rată de transfer diferite de rețelele publice – care sunt extinse spațial. O clasificare uzuală se prezintă mai jos

i) *Rețele foarte restrânse* „cluster” sunt interconectări prin medii de transfer cu lungimi de la centimetri la zeci de metri, în general localizată într-o singură încăpere. Mediile şi tehnicile folosite permit rate foarte mari de transfer iar aplicațiile vizate sunt în general cele de prelucrare paralelă a datelor.
ii) *Rețele locale* LAN („Local Area Network”) sunt interconectări de calculatoare la nivelul departamentului întreprinderii (de exemplu: serviciul Aprovizionare, serviciul Tehnic etc.). Acestea au extinderi de până la 10 km şi prezintă rate mari de transfer, tehnici de acces prin difuzare şi câteva tehnologii (implementări ale tehnicilor) care s-au impus ca standarde.
iii) *Rețele metropolitane* MAN („Metropolitan Area Network”) se extind la nivelul unui oraş, de la 10 la 100 km. Se folosesc în comun medii de transfer pentru diferite categorii de comunicații (de exemplu cabluri de înaltă frecvență pentru televiziune) astfel ca transferul să se facă direct de la sursă la destinație - prin difuzare, cu tehnici asemănătoare celor din LAN.
iv) *Rețele largi* WAN („Wide Area Network”) sunt rețele care integrează tehnici de acces şi producători diferiți (sunt rețele eterogene), cu extindere peste 100 km. WAN sunt ori rețele publice de date asemănătoare rețelelor de distribuție telefonice (sau poştale) ori rețele private – de obicei ale unor firme multinaționale. Interconectarea între rețele se face prin linii închiriate sau comutate prin operatori de telecomunicații (companii de tip Telecom - de stat şi particulare în Europa, sau companii particulare în SUA).
v) *Rețele globale* GAN („Global Area Network”) sunt interconectări de rețele între continente, folosind comunicații prin satelit sau cabluri transoceanice, care acoperă întreg globul pământesc cu o rețea de calculatoare. Astfel de rețea este Internet (rețeaua de rețele) care, în fapt, este o rețea de servicii şi se bazează pe infrastructura de comunicație a diferitor firme.

Rețelele locale (LAN) sunt astăzi obişnuite în orice tip de organizație (firmă mare sau mică, instituție de educație sau administrație), oferind un mijloc de comunicație rapid şi complet, pentru lucrul obişnuit sau pentru teleconferințe. În România au apărut companii cu caracter privat ce asigură interconectare MAN şi WAN – prin infrastructura proprie şi cu servicii proprii, oferind comunicații pentru sedii diferite ale aceleiaşi firme, legături între firme sau conectare la Internet. Pentru integrarea comunicațiilor prin imagine (TV), voce şi date se dezvoltă o infrastructură de *comunicație de bandă largă* („broadband”) care, pe lângă viteza mare de transfer, va permite funcționarea aplicațiilor multimedia în informare, educație şi comerț.

6.1.3 Comunicații prin rețele de calculatoare

Transferul de date în rețele de calculatoare presupune suportul fizic (echipamente şi medii de comunicație) dar şi programe prin care se asigură controlul comunicației şi găsirea destinațiilor şi prezentarea comodă a informațiilor la utilizatori.

Făcând o comparație cu serviciile poştale se poate considera că, lângă clădirile şi mijloacele de transport ale poştei, sunt necesare persoane care să realizeze înscrierea expeditorului şi destinației pe plicuri, cartarea şi distribuirea plicurilor pe destinații, efectuarea de servicii speciale la cerere (transport rapid sau peste rând, confirmare de primire), gestiunea codurilor poştale, apoi controlul şi rezolvarea plicurilor rătăcite, etc. Aceste operațiuni se realizează în serviciile poştale de către personalul uman, dar pentru calculatoare asemenea acțiuni se efectuează de către software.

6.1.3.1 Resurse accesibile în rețea

Toate mijloacele fizice (hardware) şi logice (software) folosite în comun de mai mulți utilizatori constituie resurse ale rețelei. Tipuri de resurse uzuale sunt:

- *fişiere şi directoare* pe discuri (fixe, optice, etc.), montate la server, la stațiile de lucru, sau în arii de discuri conectate direct la rețea şi accesibile prin intermediul unui *sistem de fişiere* (cum sunt NTFS – MS Windows, NFS – SUN, Linux / UNIX);
- *imprimante* folosite în comun de utilizatori, eventual *imprimante de rețea* – care sunt conectate direct la rețea şi nu prin intermediul unui calculator;
- *programe server*: server de fişiere (de acces la fişiere folosite în comun), server de poştă electronică (de acces emisie/recepție mesaje text), server web (de găzduire pagini web), server FTP (de transfer fişiere);
- diverse *periferice speciale*: mese de desen, interfețe de comunicație, echipamente multimedia, interfețe de proces.

Fiind folosite în comun, resursele trebuie *gestionate* adică alocate pe rând fiecărui utilizator ce solicită accesul resursei; de obicei, sarcina de gestionare a unei resurse revine unui program server (care lucrează permanent – este „rezident" în memoria de lucru) pe calculatorul sau la dispozitivul unde se află resursa (de exemplu, imprimanta de rețea are programul server instalat în memoria sa). Resursele trebuie apoi *administrate* adică întreținute periodic (verificate şi eliminate erori minore sau informații inutile), dezvoltate (perfecționate sau completate cu noi facilități), asigurate (protejate împotriva defectelor şi accesului neautorizat).

O resursă care se doreşte folosită în comun – partajată („shared") în rețea, se declară ca atare printr-o opțiune de meniu, şi se indică apoi pentru fiecare utilizator modul de acces la resursă.

6.1.3.2 Utilizatori şi drepturi de acces la resurse

Lucrul în rețea a mai multor utilizatori presupune comunicație între aceştia, folosirea în comun a unor resurse dar şi posibilitatea de a se perturba reciproc – prin ştergerea voită sau accidentală a unor fişiere, prin acces neautorizat la anumite informații, etc. Pentru evitarea acestor neajunsuri, utilizatorii se *conectează* („logon", „login") la sistemul de calcul în rețea, indicând următoarele informații proprii de anunțare şi certificare a identității:

- *nume de utilizator* („user name") – ce identifică persoana în lista de utilizatori admişi la sistem şi permite acces acesteia la *contul* personal (de obicei numele este prenumele persoanei sau o prescurtare a numelui);
- *parola* („password") – care certifică identitatea utilizatorului, este secretă şi cuprinde o combinație de caractere (litere mari/mici, cifre) greu de ghicit (recomandat a fi schimbată din când în când);
- *grupul de utilizatori* („user group" / „domain") – autorizează accesul utilizatorului la un lot de resurse (fizice – ca maşini în rețea, sau logice – ca fişiere şi directoare).

Conform grupului din care face parte, utilizatorul are un set de drepturi de acces la resursele disponibile în rețea – numite şi drepturi de încredere („trustee rights") sau permisiuni („permissions"). Aceste drepturi indică pentru fiecare resursă sau lot de resurse operațiile permise utilizatorului, de exemplu pentru fişiere: „numai citire", „citire şi scriere", „modificare drepturi de acces". Un utilizator poate indica, drepturi pe care la acordă altor utilizatori la resursele proprii - de obicei diferențiat după grupul din care fac parte („din grupul propriu" şi „alții").

Există următoarele tipuri generice de utilizatori (clasificare după drepturile de încredere şi după atribuțiile în funcționarea rețelei):

- utilizator obişnuit („user") – persoană înscrisă în registrele sistemului, care utilizează doar resursele permise, în perioadele de timp permise şi la nivelul permis (privind volumul de date transferat, spațiul de stocare alocat, etc.);
- manager de grup („group manager") – persoană desemnată din cadrul grupului, care face operațiuni de gestionare a resurselor proprii ale grupului şi ajută pe ceilalți utilizatori la rezolvarea unor incidente simple;
- administrator de rețea („superuser", „network administrator") – persoană care are ca atribuții de lucru gestionarea tuturor resurselor rețelei şi menținerea acesteia în bună funcționare.

Între aceşti utilizatori doar *administratorul de rețea* este obligatoriu cu calificare în domeniul Informaticii, el fiind răspunzător de bunul mers al rețelei (starea de funcționare, tipul şi disponibilitatea resurselor, dezvoltarea rețelei) dar şi de administrarea eficientă a resurselor şi echipamentelor de interconectare în rețea. Pentru aceasta, el deține drepturi de acces complete la toate resursele, fiind totodată primul răspunzător de integritatea şi confidențialitatea lor.

Unele resurse au ataşate *atribute* ce indică modul de acces şi starea lor, pentru a le proteja de manevre greşite ce pot veni din partea oricărui tip de utilizator (simplu sau administrator) ori pentru a efectua salvarea de siguranță („backup") a acestora pe suport extern (benzi magnetice, CD).

6.1.3.3 Tipuri de servicii în rețele locale şi largi

De obicei, accesul la o resursă partajată poartă numele de *serviciu* în rețea (acordat unui utilizator). De obicei, serviciile în rețea sunt realizate prin arhitectura client-server (două piese software legate: una pe stația de lucru alta pe server – v. §4.2.2.3). După locul unde se află resursa şi după modul de „proprietate" asupra ei, serviciul (accesul la resursă) pot fi:

- orientat server – în care resursa este „publică", se află pe o maşină dedicată care gestionează resursa şi care o alocă pe rând utilizatorilor;
- de-la-egal-la-egal („peer-to-peer") – în care resursa este „proprietatea" unui utilizator, se află pe stația sa de lucru şi este oferită de acesta spre a fi partajată („shared") cu alți utilizatori prin rețea.

Serviciile sunt oferite (şi gestionate) de programe care fac parte direct din sistemul de operare sau sunt programe separate – ce sunt lansate şi rămân rezidente în memoria de lucru, aşteptând cereri de la utilizatori şi oferind date şi/sau prelucrări specifice.

În rețele locale (LAN) sistemul de operare oferă mai ales servicii de acces la fişiere şi la periferice; serviciile sunt orientate server sau, cel mai adesea, sunt de-la-egal-la-egal, caz în care resursele puse în comun de către utilizatori sunt montate pe stațiile de lucru proprii.

În rețele largi (WAN) se instalează peste sistemul de operare programe server – fiecare pentru un serviciu anume, acesta fiind accesat de la distanță prin intermediul unei adrese unice în toată rețeaua largă (spre exemplu în Internet adresa IP şi numărul de port). Serviciile uzuale în rețele largi vizează schimbul de informații, mai rar fiind întâlnite servicii de partajare a

resurselor sau lucru la distanţă. Aceste servicii sunt realizate în arhitectură client-server, cu programul server aflat la o *locaţie* („site") cu adresa unică în reţeaua largă.

Exemple de servicii în reţeaua Internet sunt: căutare şi afişare de informaţii sub formă de documente (pagini web sau documente scrise în format electronic), tranzacţii comerciale sau financiare prin Internet, vot prin Internet, discuţii în grupuri cu moderator.

6.2 Internet şi Intranet

La începuturile sale, Internet-ul a fost o reţea experimentală finanţată de Departamentul Apărării a SUA şi dezvoltată de câteva universităţi americane, în scopul realizării unei comunicaţii de date fiabile şi care să funcţioneze şi dacă este distrusă parţial. Ideea de bază este aceea de a se înlocui comunicaţia „tradiţională" – prin „centrale" (telefonice) şi linii către fiecare abonat (în stea), cu o reţea de dispozitive de tip dispecer (dirijoare – „routers") plasate într-o reţea de tip plasă. În prima abordare, dacă se distruge centrala se blochează orice comunicaţie, pe când în a doua abordare, dacă se distruge o parte din reţea restul poate funcţiona şi chiar în zona atacată datele pot sosi pe alte căi disponibile încă în reţea prin redundanţa lor. Prin faptul că această reţea a fost gratuită la începuturi şi prin posibilităţile de comunicaţie oferite (schimb de mesaje, fişiere, aplicaţii la distanţă) ea s-a extins repede.

Astăzi, reţeaua globală Internet vine cu o deschidere nemaiîntâlnită spre comunicare, fiind mediul de convergenţă a informaţiilor din aproape toate mediile de activitate şi către orice utilităţi umane. Internet-ul a valorizat comunicarea totală, fiindcă a permis - prin caracterul său deschis şi non-profit, ca oricine doreşte să ofere şi să consume informaţii să aibă această posibilitate. Internet-ul a devenit o „piaţă" în care fiecare poate, de bună voie, să etaleze şi respectiv să găsească produse de tip „informaţie". Pe această piaţă virtuală se pot astăzi vinde şi cumpăra produse reale - fiindcă orice tranzacţie este întâi de toate un schimb de informaţii iar acest schimb se poate face între persoane aflate în orice colţ al lumii. Evident, ca şi o piaţă reală, Internet-ul este nestructurat şi uneori acoperit de „zgomot". De aceea există şi grupuri de interes (ca nişte magazine specializate sau ca nişte cluburi), cu deschidere mai mult sau mai puţin liberă publicului, prin care se poate focaliza interesul şi registrul de căutare după preocupările membrilor grupului.

6.2.1 Structura Internet-ului

Reţeaua Internet are o structură ierarhică, dezvoltată în jurul unui „miez" („Internet core") drept coloană vertebrală („backbone") a întregii reţele dar cu magistrala circulară.

În Internet se folosesc denumiri de domenii care desemnează zone geografice (ţări) sau sectoare de ocupaţie, având fiecare la rândul lor o structură ierarhică în care numele desemnează organizaţia (ca instituţie comercială, de administraţie, de educaţie, etc.). Exemple de numele de domenii sunt:

- Tipuri de organizaţii: com sau co – comerciale, edu sau ac – educaţie, org – organizaţii non-profit, net – reţea sau furnizori de servicii în reţea, gov – guvern, biz – afaceri, coop – cooperaţii, name – indivizi;
- Zone geografice: fr – Franţa, ro – Romania, uk – Anglia, nl – Olanda.

Administrarea globală a adreselor de identificare a domeniilor se face de către forul denumit NIC (Network Information Center) iar conducerea (tehnică) a Internet este efectuată de forul IETF (Internet Executive Task Force).

6.2.1.1 Internet şi WWW

În 1989 Tim Berners Lee de la Laboratorul European de Fizica Particulelor din Geneva (CERN) a propus un sistem de informare bazat pe hipertext pentru echipele de cercetători,

acestea fiind situate în zone geografice diferite, folosind calculatoare diferite şi documente realizate în tehnologii deferite. Scopul sistemului era de a oferi o interfață standard pentru operare şi prezentarea documentelor, precum şi şi posibilitatea de integra diferite tipuri de servicii (existente la acel moment în Internet). Sistemul a fost denumit „World Wide Web" (rețea largă cât lumea), prescurtat www sau web. Această soluție de prezentare standard şi de integrare a serviciilor a revoluționat Internet-ul, astfel că astăzi – când se discută de Internet se subînțelege în cele mai multe cazuri www.

Hipertextul („hypertext") este un text electronic „în trei dimensiuni": pe lângă organizarea pe linii care se succed în *pagină*, există *legături* („hyperlinks", „links") cu alte pagini aflate pe maşina curentă sau pe o maşină distantă. Legăturile sunt de fapt trimiteri (ce se regăsesc şi în cărțile tipărite) care însă în pagina electronică permit „saltul" la informația indicată şi vizualizarea documentului pe loc. Limbajul de descriere a paginii web se numeşte HTML (Hyper Text Markup Language) şi are caracteristic faptul că separă conținutul documentului (text şi imagini) de informația de formatare.

Paginile web reprezintă documente de informare către utilizator, cu facilități de acces interactiv în preluarea de date de la utilizator. Pagina web şi mijloacele de interacțiune sunt găzduite de o maşină într-o locație anume („site") pe glob. Publicarea paginii web se face prin intermediul unui program server web, iar utilizatorul poate accesa şi vizualiza pagina de la calculatorul propriu – prin intermediul unui program client web denumit navigator („browser" – „răsfoitor").

Utilizarea web-ului constă în „navigarea" prin intermediul adreselor web şi a legăturilor din hipertextul aflat în paginile web, pentru vizualizarea şi, eventual, furnizarea informațiilor de interes. Serviciile oferite de web sunt foarte diverse, însă interfața generală de lucru şi modul de navigare este standard.

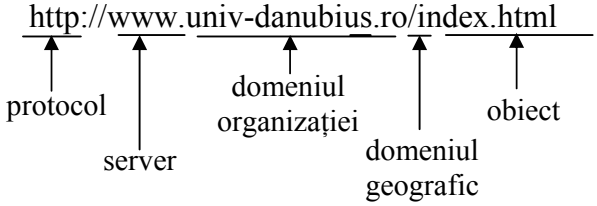
6.2.1.2 Adrese Internet

Pentru a fi accesată, o maşină conectată la rețeaua de calculatoare trebuie să aibă alocată o adresă unică în rețea, astfel ca la emisia datelor să indice destinația fără echivoc iar la recepție datele să fie îndrumate către ea prin sistemele intermediare. În Internet, adresele sunt specificate prin numere – „patru numere cu punct" în IPv4 (sau cu 16 numere pentru IPv6 în viitor), de exemplu 192.168.11.33. Este dificil pentru om să rețină aceste numere; în locul lor se folosesc cuvinte, care au semnificație pentru om şi sunt deci uşor de memorat şi utilizat.

Exprimarea locului şi obiectului ce se doreşte accesat în Internet se face prin URL („uniform resource locator") care este o înşiruire de cuvinte separate cu puncte, fiecare cuvânt exprimând numele de domeniu, organizație, maşină şi obiect – în ordinea invers ierarhică a acestora. Structura generică a URL – denumit pe comun „adresă Internet" este:

<protocol>://<maşină.organizație.domeniu>:<port>/<obiect>

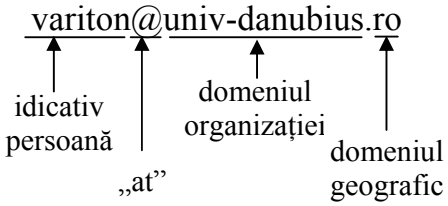
unde <protocol> indică tipul aplicației de acces la date (şi convenția sa de lucru) iar <port> adresa pe maşina locală a programului (folosită în unele cazuri), ca în exemplul:



prin care se accesează pagina de prezentare a universității „Danubius" din Galați. Pe acest exemplu se indică rolul fiecărui cuvânt din structura unui URL.

Fiindcă WWW integrează diverse tipuri de aplicații, fiecare din acestea poate fi accesată indicând protocolul specific ei; de exemplu, aplicația de transfer fişiere pe un site foarte cunoscut este ftp://ftp.funet.fi. – în care s-a indicat protocolul şi serverul FTP de la „rețeaua academică finlandeză".

În cazul în care adresa Internet se referă la o căsuță poştală („e-mail box"), adresa va conține un indicativ (nume) al persoanei sau departamentului proprietare al căsuței poştale:



Simbolul @ se pronunță ‚la' („at" în engleză) indicând locația unde se află contul de poştă electronică a persoanei; deseori acest simbol este indicat prin expresia „a rond" când se citează adresa de poştă electronică.

6.2.2 Aplicații uzuale în Internet

Deseori în activitatea umană apar situații care solicită comunicații la distanță; în asemenea cazuri, se pot folosi aplicații în Internet, ce permit:

a. *navigare şi vizualizare* informații pe *pagini web* (hipertext);
b. *căutare după cuvinte cheie* a documentelor sau locațiilor cu anumite informații de interes (folosind „maşini de căutare");
c. *căutarea de persoane sau instituții* după nume sau după organizația din care face parte;
d. *operațiuni efectuate de la distanță* – pentru comerț, tranzacții financiare, dezvoltare/ up-grade de aplicații;
e. *transfer de fişiere la distanță* de la un server de fişiere („down-load") sau către acesta („up-load") pentru utilizatori înregistrați sau ca vizitator („guest", „anonymous");
f. *lucru la distanță* prin care se pot da comenzi şi datele de prelucrare de pe maşina proprie către o maşină distantă (de obicei mai puternică) – ca şi cum utilizatorul s-ar afla la consola acesteia din urmă, folosind comenzi ale sistemului ei de operare;
g. *mesagerie electronică* („e-mail") prin care se transmit mesaje text însoțite de imagini şi fişiere de date – direct în mesaj (MHS sau X-400) ori ataşate (poşta electronică SMTP);
h. *grupuri de discuții* pe teme de interes comun mai multor persoane, în care este permis accesul la informații specializate sau cu circulație restrânsă şi au loc dezbateri la distanță, între persoane ce s-au înregistrat fiecare ca membru al grupului.

Diverse alte servicii – cum sunt educație de la distanță („e-learning"), teleconferințe (video sau doar prin voce şi date), informații şi acțiuni administrative sau vot la distanță („e-government"), acțiuni de informare şi ajutor în sănătate („e-health"), se bazează pe combinații ale serviciilor de mai sus şi în special pe navigare („browsing"). De obicei, serviciile sunt oferite de un program server aflat într-o *locație cunoscută* („site"), în conexiune cu alte locații.

6.2.3 Conectarea la Internet

Un calculator poate folosi servicii în Internet dacă este „conectat la Internet", adică este inclus în rețeaua largă prin *infrastructura de comunicație* şi apoi are alocată o adresă Internet (adresă IP) pentru a fi accesat în rețea prin intermediul unui *Furnizor de Servicii Internet* (ISP – „Internet service provider"). Conectarea unui calculator la rețeaua largă şi la furnizorul de servicii se poate face în diverse moduri, după cum se prezintă în continuare.

6.2.3.1 Conectare prin modem la linia telefonică

La transportul pe linii telefonice a semnalelor digitale (impulsuri) acestea sunt deformate şi comunicaţia nu poate avea loc; de aceea, semnalul digital se „încarcă" peste un semnal analogic prin modulare (la emisie) şi se „descarcă" prin demodulare la recepţie. Dispozitivul care face modularea şi demodularea se numeşte *modem de linie telefonică*. Constructiv, modemul poate fi modem intern – ca placă ce se montează pe magistrala sistemului de calcul (ISA sau PCI – v. §1.3.1.3), sau poate fi modem extern – ca dispozitiv independent conectat la interfaţa serială RS232 sau la magistrala USB a sistemului de calcul.

Un sistem de operare modern permite depistarea modemului şi includerea sa în setul de periferice al sistemului de calcul, printr-un program „driver" ce controlează funcţionarea modemului. Debitul de comunicaţie la conectarea prin modem de linie telefonică poate fi de 14 Kbps până la 56 Kbps (Kilo bit pe secundă).

Conectarea modemului la furnizorul de servicii se poate face prin *linie comutată* (prin centrala telefonică) sau prin *linie închiriată* – linie directă folosită numai pentru transferul de date al beneficiarului. Evident, costurile diferă dar viteza de transfer este mult mai mare în cazul folosirii liniilor închiriate.

6.2.3.2 Conectare la linii de bandă largă

Aplicaţiile actuale prin care se oferă servicii în Internt necesită „bandă largă" („broadband") adică un debit mare la transferul de date (impropriu considerate „viteze mari"). Furnizorii de servicii Internet oferă infrastructura de comunicaţie şi dispozitive pentru a asigura aceste debite (viteze) mari, cu una din tehnologiile:

- *ISDN* (Integrated Services Digital Network) – ca infrastructură de mare viteză care digitizează şi mixează comunicaţii de date, voce (telefon), fax, video pe aceleaşi linii. Când serviciul este oferit de furnizori de telefonie digitală se prevăd două canale a 64 Kbps de date şi unul de 16 Kbps de control – astfel că rata maximă de transfer este de 144 Kbps, dar când serviciul este oferit prin fibră optică poate ajunge până la 2 Mbps (pe baza a maxim 30 canale de date).
- *ATM* (Asynchronous Transfer Mode) – denumit şi B-ISDN, permite telecomunicaţii de bandă largă (de 622 Mbps dar poate ajunge şi până la 2,48 Gbps) mixând date, voce şi imagine digitizate. Transmisia se face prin pachete mici de date, (53 octeţi) în reţele comutate (pe principiul centralelor telefonice), totdeauna la viteza maximă, prin fibră optică, local (în interiorul organizaţiei) sau la mare distanţă.
- *Bandă largă prin CATV* (Community Antenna Television) – transfer de date prin cablu TV la furnizori de televiziune prin cablu, prin cablu coaxial (pe canale de 6 MHz lărgime de bandă) sau prin fibră optică (DQDB de la 50 Mbps la 600 Mbps) în reţele MAN.

Pentru fiecare categorie de conectare este necesar un modem special şi plata serviciului.

6.2.3.3 Conectare prin reţeaua locală

Atunci când organizaţia dispune de o reţea (ca reţea locală LAN – dacă organizaţia este mică, sau interconectare de reţele locale – la organizaţii mari) fiecare calculator este conectat la mediul de transfer în LAN prin intermediul unei *plăci de reţea* (sau „adaptor/controller de reţea"). Mediul de transfer este de obicei cuprul - prin cablu torsadat ne-ecranat (UTP), iar reţeaua locală prezintă echipamente de interconectare (v. §6.1.1.3), de obicei prin „switch".

Uzual, reţelele locale sunt de tip Ethernet (pe cablu UTP) având rate de transfer de 10 Mbps sau 100 Mbps. Efectiv, rata de transfer în Internet este însă determinată de *tipul de acces la*

Internet a organizației: prin linie telefonică (comutată sau închiriată) sau prin linie de bandă largă (fibră optică şi servicii de bandă largă – v. §6.2.3.2).

Dependent de tehnologia rețelei locale, placa de rețea prezintă o adresă fizică (pentru nivelurile 1 şi 2) prin care este accesată în rețeaua locală, precum şi o adresă Internet (adresă IP – pentru nivelul 3) prin care este accesată în Internet. Adresa fizică este, de obicei, înscrisă fix în placa de rețea iar adresa IP este înscrisă de administratorul de rețea în sistemul de operare sau este furnizată pe loc – la conectarea în Internet, de către un server specializat (RARP). Pentru conectarea organizației la Internet, este necesar un echipament de interconectare de tip *dirijor* („router"), eventual fiind emulat prin sistemul de operare al maşinii server pentru Internet (web, e-mail, etc.) al organizației.

6.2.4 Intranet

Prin Intranet se înțelege aplicarea tehnicilor şi conceptelor *Internet în interiorul unei organizații*, folosind protocoale TCP/IP şi tehnologii WWW. Astfel Intranet devine un Internet privat în care utilizatorii pot folosi navigatoare şi celelalte facilități de comunicare WWW pentru a schimba informații între ei. În plus, Intranet-ul este separat de restul rețelei Internet prin sisteme de *protecție de tip parafoc* („firewall"), prin care se urmăresc toate operațiile de acces din afară către Intranet şi se blochează operații nepermise sau care reprezintă un pericol pentru datele şi programele organizației.

Cum astăzi se tinde spre o integrare a tuturor tipurilor de comunicații, Intranet permite utilizarea în interiorul organizației a platformei universale de comunicație (multimedia şi hipermedia) pe care navigatoarele web le pun la dispoziție. În fapt, este necesar ca în fiecare loc din mediul organizațional Intranet, de unde se doreşte să se facă publice informații, să existe un server web, la care se pot conecta prin clienți web (navigatoare ca Internet Explorer, Netscape, etc.) membrii organizației.

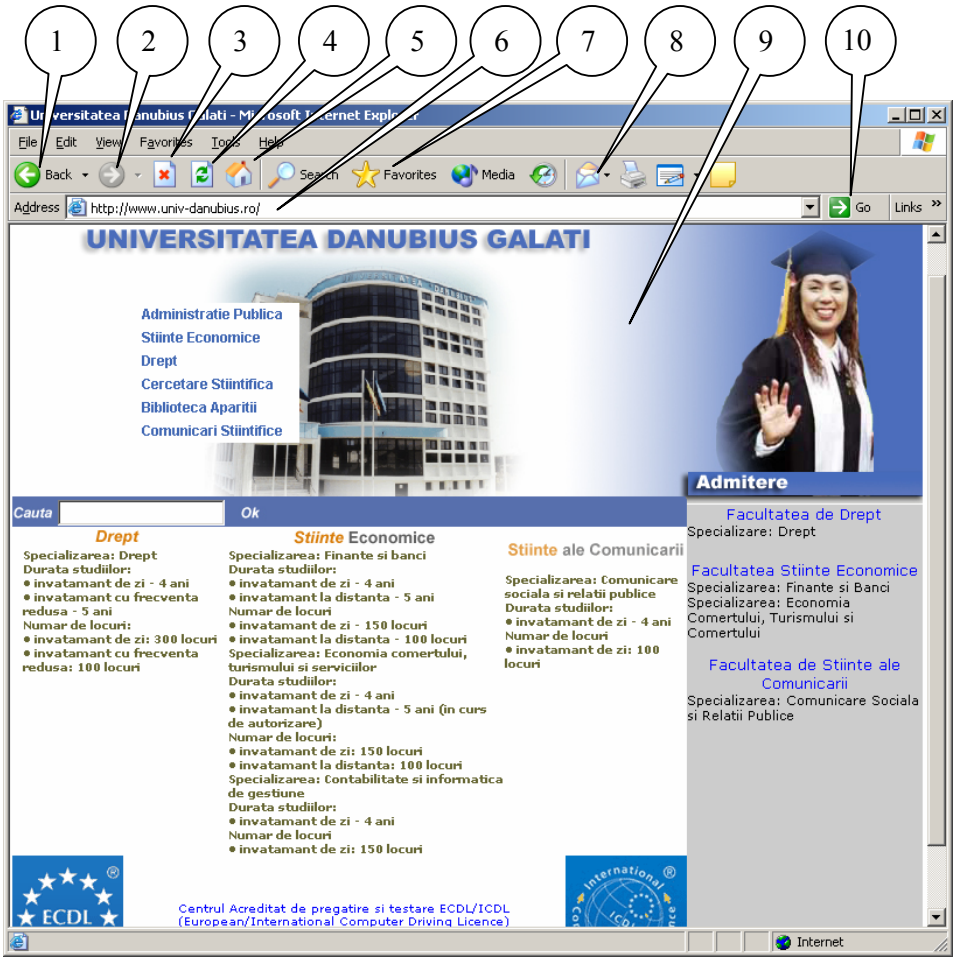
6.3 Utilizarea aplicațiilor de comunicație în Internet

Dintre aplicațiile uzuale în Internet, cele mai importante sunt navigatoarele web şi poşta electronică. Navigatorul este folosit pentru vizualizarea paginilor web, pentru interacțiunea cu servere în locații distante („site-uri"), precum şi pentru căutare în Internet. Poşta electronică (simplă - SMTP) este folosită pentru transmiterea şi recepționarea de mesaje text, eventual cu fişiere ataşate. Utilizarea acestor aplicații este posibilă dacă maşina pe care ele se află instalate este conectată la Internet (prin intermediul unui ISP) şi dacă există un server adecvat pentru fiecare. În continuare, se prezintă utilizarea părții client pentru navigator şi poşta electronică, cu exemple de implicare şi exploatare a părții server.

6.3.1 Navigator Web

Produsele de tip navigator Internet constituie partea client a aplicației de vizitare a paginilor web. După conectarea la Internet se poate accesa serverul web (denumit „site") care găzduieşte pagina cu informații de interes.

Prima pagină vizitată este numită pagina de start („home page" v. figura de mai jos) care poate fi pagina de prezentare a organizației proprii sau o altă pagină preferată. Utilizatorul va executa „salturi" spre alte locații („site"-uri) prin intermediul legăturilor oferite de hipertextul din paginile pe care le vizitează sau poate înscrie direct, în bara de adresă (6), adrese (URL) cunoscute.



O pagină web de start („home page”).

Bara de unelte a ferestrei navigatorului (de exemplu MS Internet Explorer în figura de mai jos) conține elemente necesare parcurgerii paginilor web – ce au fost vizitate sau se vor vizita: (1) „parcurgere înapoi” - vizionarea paginii precedente, (2) „parcurgere înainte” - vizionarea paginii următoare (dacă s-a revenit la una precedentă), (3) oprirea încărcării paginii cu adresa curentă, (4) reîncărcarea paginii curente, (5) salt la pagina de start, (6) casetă de text pentru înscrierea adresei URL ce se dorește vizitată, (7) colecție de adrese (URL) preferate (păstrate ca „favourites” sau semne de carte „bookmarks”), (8) acces la client de poștă electronică disponibil în navigator, (9) pagina web vizualizată, (10) buton de inițiere a saltului la pagina cu adresa indicată în (6).

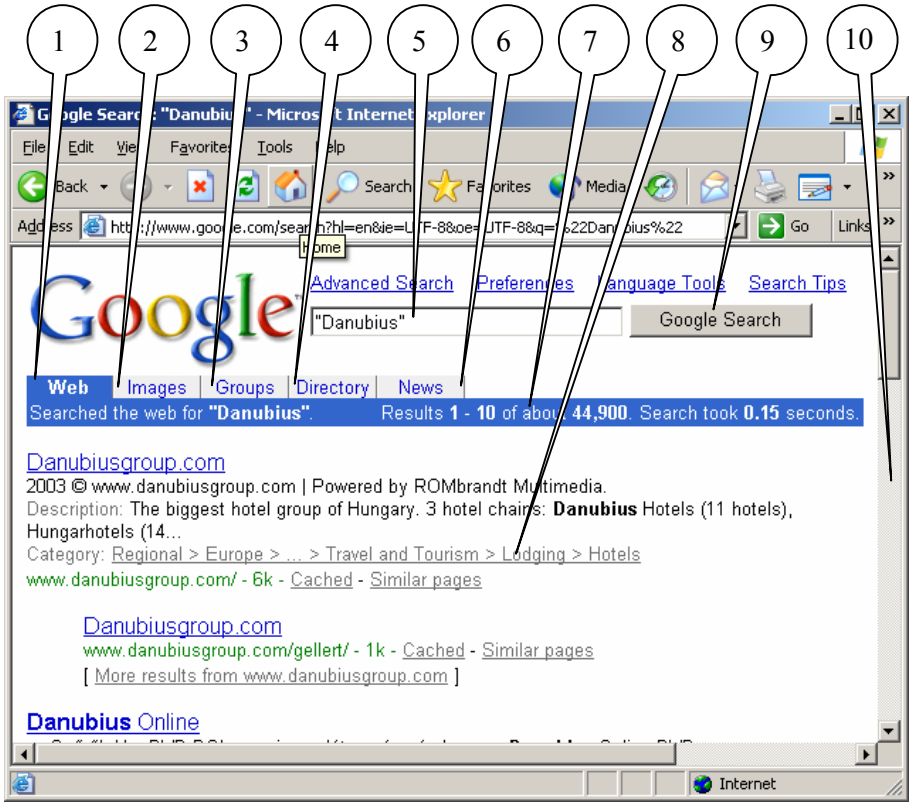
Pagina curentă se poate salva și tipări (cu opțiuni din meniul „File”), se poate declara ca pagină de start (în meniul „Tools-Internet Options”), se poate vizualiza în formatul sursă al limbajului de descriere HTML (opțiune în meniul „Edit”), se poate include între paginile preferate (opțiune în meniul „Favorites”).

În general, o pagină web prezintă chiar în conținutul său hiperlegături de navigare: „parcurgere înapoi” și „parcurgere înainte”, salt la încput și salt la sfârșitul paginii. Aceste facilități și hiperlegăturile din pagină dau utilizatorului senzația de libertate de alegere a căilor și obiectelor de vizitare în „spațiul cibernetic” – cum este denumit Internet-ul.

6.3.2 Maşini de căutare

Navigarea fără o țintă precisă se reduce la „hoinăreală” – bună eventual pentru distracție nu pentru realizarea unui acțiuni. De aceea, există în Internet „motoare de căutare” care au rolul de a reduce spațiul de parcurs la o acțiune de informare. Un motor de căutare este o interfață interactivă ce permite *formularea de interogări către o bază de date* care conține cuvinte cheie şi legături spre locații ce dețin documente unde apar aceste cuvinte cheie.

Interfața interactivă prezintă obligatoriu o casetă de text pentru introducerea interogării - sub forma cuvântului sau combinației de cuvinte de căutat. În figură se indică elementele uzuale ale interfeței unei maşini de căutare, prin exemplificare pe maşina www.google.com:



Pagină cu rezultate ale căutării prin maşina Google.

(1), (2), (3), (4), şi (6) *fişe cu categorii* de elemente țintă căutate, în ordine: pagini Web, Imagini, Grupuri (de interes pe domenii), Cataloage (structuri de directoare şi subdirectoare ce organizează informația), Ştiri (grupuri de ştiri). Conținutul în informații din categoria selectată se afişează mai jos în pagină.

(5) *caseta de formulare a interogării*, în care se înscriu cuvintele cheie (incluse, eventual, în expresii logice - v. mai jos operatori).

(7) *informații sintetice* privind rezultatele căutării: numărul de rezultate afişate, totalul acestora, timpul de răspuns.

(9) *butonul de start* al căutării – acționat după descrierea completă a expresiei de căutare.

(8) *element găsit*, în categoria fişei selectate, care este indicat prin hiperlegătura către locația care apare, prezentarea scurtă a contextului şi legăturile ierarhice specifice locației țintă.

(10) bară de derulare a setului de elemente găsite şi afişate în panoul curent.

De obicei, în josul paginii afişate, se găsesc butoane de navigare către pagini ulterioare (şi precedente) în mulțimea de elemente afişate.

Operatorii folosiți în expresii de interogare complexe - pentru rafinarea căutării, pot fi:

- „” cuvinte între ghilimele înseamnă că pagina trebuie să conțină exact fraza dintre ghilimele
- + cuvântul care are plus înainte (ex. +universitate) trebuie neapărat să existe în pagină
- - cuvântul care are minus înainte (ex. –hotel) nu trebuie să se găsească în pagină (echivalent cu NOT)
- AND – de exemplu: danubius AND galati (prin care se caută pagini care conțin obligatoriu ambele cuvinte); cuvintele pot fi scrise cu inițiale majuscule sau nu.
- OR – de exemplu: danubius OR universitate (prin care se caută pagini care conțin măcar unul din cuvinte sau amândouă).

Se fac următoarele recomandări pentru o căutare eficientă:

i) Se specifică cît mai simplu şi direct interogarea (indicând cuvântul căutat).
ii) În cazul unui set mare de elemente rezultat se analizează aspectele efectiv dorite pentru informația căutată şi se precizează sinonime sau antonime.
iii) Se creează expresii de interogare (folosind operatori), care să includă sau să excludă cuvinte specifice aspectelor vizate (sinonime, respectiv antonime).
iv) Se parcurg măcar 50 de elemente din primele pagini rezultat al interogării.
v) Se parcurg pagini similare sugerate de maşina de căutare.

Există diverse locații care oferă maşini de căutare cu diferite facilități: www.google.com (oferă expresii eficiente de restrângere a căutării), www.lycos.com (oferă cataloage organizate după subiecte), www.infoseek.com (oferă cu catalog extensiv de locații clasificate după subiect), www.yahoo.com (mai curând un catalog cu clasificarea locațiilor pe domenii).

6.4 Rezumat

Comunicația este integrată astăzi domeniului Tehnologiei Informației, pentru că ea utilizează aceleaşi mijloace ca şi prelucrarea datelor. Rețelele de calculatoare sunt structuri de echipamente interconectate prin diverse medii de transfer ale undelor electromagnetice (cable de cupru, fibră optică, eter), care servesc utilizatorul prin intemediul unor calculatoare plase central (servere), care trasnmit date prelucrate la pe maşina utilizatorului (maşina „client”). Astfel, sunt disponibile utilizatorilor conectați în rețea diverse resurse partajate (adică puse în comun), însă fiecare poate uza de resurse doar în limitele permisisunilor stabilite de către administratorul de rețea. Aceste permisiuni au rolul de asigura: autorizarea accesului, securitatea datelor importante, confidențialitatea datelor sensibile, administrarea comodă a unui mare volum de date şi programe. Internetul este un exemplu de rețea de calculatoare care însă diferă de rețelele locale (LAN) sau cele metropolitane (MAN) prin extinderea sa globală şi prin accesul liber la resurse de informare şi de efectuare tranzacții comerciale. Organizațiile pot crea intranetul propriu – ca o replică în mic a Internetului, prin care angajații se pot informa şi pot comunica eficient.

6.5 Teme de control

1. Denumiți echipamente de calcul ce servesc utilizatorul uman în rețele de calculatoare, cu rolurile lor.
2. Care sunt diferențele de interconectare a două procesoare prin cuplaj strâns şi prin cuplaj slab?
3. Ce diferențe şi ce asemănări găsiți între rețele locale şi cele metropolitane?
4. Enumerați resurse uzuale accesibile în rețea, cu indicarea scurtă a specificului lor.
5. Ce informații trebuie să furnizeze utilizatorul la conectarea în rețea?

6. Indicați trei atribuții ale administratorului de rețea, şi comentați necesitatea acestora.
7. Prin ce se aseamănă şi prin ce se deoseesc Internet-ul şi intranet-ul?
8. Indicați şi explicați trei prescurtări folosite ca denumiri de domenii în Internet.
9. Ce este o hiperlegătură şi ce este un hipertext?
10. Cum se numeşte generic programul care afişează pagini WWW şi de unde se accesează aceste pagini?
11. Ce rol are simbolul @ în cadrul unei adrese de poştă electronică?
12. Indicați două tipuri de servicii prin care pot comunica direct două persoane prin Internet.
13. Prin ce se deosebesc conectarea la Internet folosind linia telefonică de cea folosind o rețea locală?
14. Ce este „home page" şi ce este URL indicate prin intermediul unui navigator WWW?
15. Dați exemplu de o expresie de căutare cu două cuvinte cheie, folosită în maşina de căutare Google.

Exemple de răspuns:

(3) Atât rețelele locale cât şi cele metropolitane permit rate mari de transfer şi tehnici de acces prin difuzare, dar rețelele locale se extind doar până la nivelul unei clădiri, în timp ce rețelele motropolitane se extind până la nivelului unui oraş şi folosesc uneori infrastructuri de televiziune prin cablu deja montate.

(11) Simbolul @ separă numele de utilizator al căsuței poştale de numele subdomeniilor şi domeniului unde această căsuță este rezidentă.

Bibliografie

Ariton V., *Reţele de calculatoare*, Editura Evrika Galati, 1999.

Bott E., Leonhard W., *Microsoft Office XP*, Teora, 2002.

Brookshear J. G., *Introducere în Informatică*, Ed. Teora, 1998.

Burdescu D. D., *Algoritmi şi structuri de date*, Ed. Mirton, 1992

Cowart R., Knittel B., *Microsoft Windows Professional*, Teora, 2002

Funeriu I., *Principii şi norme de tehnoredactare computerizată*, Editura Amarcord, 1998.

Gralla P., *How Intranets Work*, ZD PRESS Macmillan Computer Publishing, 1998.

Jacobson I., Ericsson M. and Jacobson A., *The Object Advantage: business process engineering with object technology*, Addison-Wesley, 2001.

Odăgescu I., Smeurean I., Ştefănescu I., *Programarea avansată a calculatoarelor personale*, Ed. Militară 1993.

Rumbaugh J., et al., *Object Oriented modeling and design*, Prentice Hall Int., Englewood Cliffs, 1991.

Scholtz-Reiter B., et al., *Business Process Modelling*, Heidelberg, Springer, 1996.

Schreiber, G., Wielinga, B. and Breuker, J., *KADS: A Principled Approach to Knowledge-Based System Developmen*t. Academic Press, London, UK, 1993.

Somnea D., Calciu M., Dumitrescu E., *Birotica*, Ed. Tehnica, 1998

Stroustrup B., *The annotated C++ Reference manual*, Addison Wesley, 1991.

Tardieu H., et al., *La methode MERISE. Principes et outils*, Les editions d'organisation, 1986.

Wilson D. A., *Managing Information*, Butterworth-Heinemann, 1998.

Zadeh L. A., *Fuzzy sets*, Information and Control, 8:338-353, 1965.

Zorkoczy P., Heap N., *Information Technology – an introduction*, Pitman Publishing, 1995.

Cuprins

1 Introducere 1
1.1 Informații și prelucrări 1
1.1.1 Date 2
1.1.2 Prelucrarea datelor 2
1.1.3 Scopul prelucrării electronice a informațiilor 3
1.1.4 Organizarea datelor 3
1.1.5 Stocarea datelor (fișiere) 4
1.1.6 Introducerea și prezentarea datelor 4
1.2 Comunicații de date și multimedia 5
1.3 Sisteme de calcul 5
1.3.1.1 Procesorul 6
1.3.1.2 Memoria internă 6
1.3.1.3 Magistralele sistemului 7
1.3.1.4 Placa de bază 7
1.3.1.5 Unități periferice de intrare 7
1.3.1.6 Unități periferice de ieșire 7
1.3.1.7 Unități de memorare pe suport extern 8
1.3.1.8 Tipuri constructive de calculatoare 8
1.3.2 Structura logică a unui sistemului de calcul 8
1.3.3 Principiul de lucru al unui sistem de calcul 9
1.4 Rezumat 9
1.5 Teme de control 9

2 Reprezentarea și structurarea informației 11
2.1 Bit, octet și multipli 11
2.2 Identificator, variabilă, constantă, literal 11
2.3 Tipuri de date simple 12
2.3.1 Reprezentarea numerelor întregi 12
2.3.1.1 Numere binare 12
2.3.1.2 Tipuri de date întregi 13
2.3.1.3 Operații cu numere întregi 13
2.3.2 Reprezentarea numerelor reale 13
2.3.2.1 Operații cu numere reale 14
2.3.3 Tipul de date caracter 14
2.3.3.1 Reprezentarea caracterelor 14
2.3.3.2 Operații cu tipul de date caracter 14
2.3.4 Tipul de date logic 14
2.3.4.1 Operații cu tipul de date logic 15
2.3.5 Tipuri de date structurate 15
2.3.5.1 Tipul de date tablou 15
2.3.5.2 Tipul de date șir de caractere 15
2.3.5.3 Tipul de date articol 15
2.3.6 Tipuri abstracte de date – Clase de obiecte 16
2.4 Rezumat 16
2.5 Teme de control 17

3 Prelucrarea informației 18
3.1 Expresii 18
3.2 Instrucțiuni 18
3.2.1 Instrucțiuni simple 19
3.2.1.1 Instrucțiunea de atribuire 19
3.2.1.2 Instrucțiuni de salt 19
3.2.2 Instrucțiuni structurate 19
3.2.2.1 Instrucțiunea de decizie binară 20
3.2.2.2 Instrucțiunea de decizie multiplă 20
3.2.2.3 Instrucțiuni de repetiție 20
3.2.2.4 Instrucțiunea de repetiție după condiție 20
3.2.3 Programe şi subprograme 21
3.2.3.1 Subprograme 21
3.2.3.2 Programul principal 21
3.3 Algoritmi 21
3.3.1 Exprimarea algoritmilor 21
3.3.1.1 Organigrame (Scheme logice) 22
3.3.1.2 Pseudocod 22
3.3.2 Elaborarea algoritmilor 22
3.4 Categorii de prelucrare şi prezentare a informațiilor 23
3.4.1 Calcule matematice 23
3.4.2 Prelucrări de birou 23
3.4.3 Prelucrări prin metode de Inteligență Artificială (IA) 24
3.5 Rezumat 25
3.6 Teme de control 25

4 Realizarea programelor şi programe suport 27
4.1 Problematica programării 28
4.1.1 Etape în ciclul de viață ale unui produs program 28
4.1.1.1 Formularea cerințelor 29
4.1.1.2 Analiza problemei 29
4.1.1.3 Proiectarea aplicației 30
4.1.1.4 Implementarea şi testarea aplicației 31
4.1.1.5 Exploatarea şi dezvoltarea aplicației 32
4.1.2 Limbaje de programare 33
4.1.2.1 Limbaje de programare uzuale 33
4.1.2.2 Clasificări ale limbajelor de programare 35
4.1.2.3 Compilatoare şi interpretoare 36
4.1.3 Ingineria programării 36
4.2 Tehnici şi instrumente de realizare a programelor 37
4.2.1 Caracteristici ale programării orientate obiect 39
4.2.2 Tipuri de aplicații 39
4.2.2.1 Aplicații 40
4.2.2.2 Aplicații de Baze de Date 40
4.2.2.3 Aplicații client-server 40
4.2.2.4 Miniaplicații 41
4.2.3 Instrumente software de dezvoltare a aplicațiilor 41
4.2.3.1 Interfețe, biblioteci şi pachete de programare 41
4.2.3.2 Medii de programare 42
4.2.3.3 Medii pentru Baze de Date 42

4.2.3.4 Instrumente de asistare a dezvoltării software 42
4.3 Sisteme de operare şi programe utilitare 43
4.3.1.1 Funcțiile ale sistemului de operare 43
4.3.1.2 Programe utilitare 44
4.4 Interfețe utilizator 44
4.4.1 Tastatura şi mouse-ul 45
4.4.2 Ferestre şi casete de dialog 45
4.4.2.1 Suprafața de lucru 45
4.4.2.2 Ferestre aplicație 46
4.4.2.3 Casete de dialog 47
4.5 Rezumat 48
4.6 Teme de control 48

5 Utilizarea calculatoarelor pentru aplicații de birou 49
5.1 Procesoare de texte 49
5.1.1 Structura a unui document scris 50
5.1.1.1 Structura informației în document 50
5.1.1.2 Structura de conținut a documentului 51
5.1.1.3 Structura foii tipărite 51
5.1.2 Scrierea textului – caractere şi paragrafe 52
5.1.2.1 Caractere 52
5.1.2.2 Paragrafe 52
5.1.3 Liste şi tabele 53
5.1.3.1 Liste numerotate şi ne-numerotate 53
5.1.3.2 Tabele 54
5.1.3.3 Imagini şi plasarea lor în text 54
5.1.4 Verificarea şi corectarea ortografică şi gramaticală. 55
5.1.5 Lucrul cu documente 55
5.1.5.1 Manevre de Editare 55
5.1.5.2 Lucrul cu ferestre 56
5.1.5.3 Şabloane şi scrisori tip 56
5.1.6 Indicații de tehnoredactare computerizată 56
5.1.6.1 Definirea stilurilor de formatare 56
5.1.6.2 Paşi în redactarea corectă şi eficientă a documentului 57
5.2 Foi de calcul tabelar 57
5.2.1 Structura foii de calcul tabelar 57
5.2.2 Conținutul celulelor foii de calcul 59
5.2.3 Lucrul cu foaia de calcul 60
5.2.3.1 Adrese relative şi absolute 60
5.2.3.2 Funcții 60
5.2.4 Facilități de prezentare calitativă şi sintetică a informațiilor 62
5.2.4.1 Grafice 62
5.2.5 Indicații de proiectare şi construire a foilor de calcul 64
5.2.5.1 Formularea cerințelor şi analiza problemei 64
5.2.5.2 Proiectarea şi realizarea foii de calcul 64
5.3 Rezumat 65
5.4 Teme de control 65

6 Rețele de calculatoare şi comunicații 67
6.1 Problematica rețelelor de calculatoare 67
6.1.1 Componente ale rețelelor de calculatoare 68
6.1.1.1 Echipamente de prelucrare a datelor 68
6.1.1.2 Mediu de transfer 68
6.1.1.3 Echipamente de interconectare 68
6.1.2 Clasificarea rețelelor de calculatoare după extindere spațială 69
6.1.3 Comunicații prin rețele de calculatoare 70
6.1.3.1 Resurse accesibile în rețea 70
6.1.3.2 Utilizatori şi drepturi de acces la resurse 70
6.1.3.3 Tipuri de servicii în rețele locale şi largi 71
6.2 Internet şi Intranet 72
6.2.1 Structura Internet-ului 72
6.2.1.1 Internet şi WWW 72
6.2.1.2 Adrese Internet 73
6.2.2 Aplicații uzuale în Internet 74
6.2.3 Conectarea la Internet 74
6.2.3.1 Conectare prin modem la linia telefonică 75
6.2.3.2 Conectare la linii de bandă largă 75
6.2.3.3 Conectare prin rețeaua locală 75
6.2.4 Intranet 76
6.3 Utilizarea aplicațiilor de comunicație în Internet 76
6.3.1 Navigator Web 76
6.3.2 Maşini de căutare 78
6.4 Rezumat 79
6.5 Teme de control 79

Bibliografie 81