PROF. UNIV. DR. ING. AUREL ŞERB

STRUCTURI DE CALCULATOARE, SISTEME DE OPERARE ŞI PROGRAME APLICATIVE

PRO UNIVERSITARIA 2012®
BUCUREŞTI

Copyright © 2012, **Editura Pro Universitaria**

Toate drepturile asupra prezentei ediţii aparţin
Editurii Pro Universitaria

Nicio parte din acest volum nu poate fi copiată fără acordul scris al **Editurii Pro Universitaria**

Descrierea CIP a Bibliotecii Naţionale a României
ŞERB, AUREL
Structuri de calculatoare, sisteme de operare şi programe aplicative / Aurel Şerb. - Bucureşti : Pro Universitaria, 2012
Bibliogr.
ISBN 978-606-647-504-4

004(075.8)

PREFAŢĂ

A scrie o carte de informatică, care să răspundă necesităților viitorilor specialiști din domeniul administrației publice este o provocare şi un pariu cu tine însuţi. Este o provocare pentru că termenul "Informatică pentru administrația publică" este atât de vast şi de neprecizat încât eşti supus tuturor tentaţiilor legate de accentuarea unuia sau altuia dintre disciplinele punctuale care însumate constituie marea ştiinţă a calculatoarelor. Este un pariu pentru că trebuie să găseşti un echilibru între părţile acestui mare întreg. Cartea această încearcă să prezinte un punct de vedere despre ştiinţa calculatoarelor, adresându-se celor pentru care ştiinţa calculatoarelor nu va fi în viitor profesia lor de bază. Acest manual încearcă să furnizeze cititorilor atât o sumă de cunoştinţe fundamentale, care să îi ajute să înţeleagă ce este important în domeniul ştiinţei calculatoarelor, cât şi o altă parte foarte concretă legată punctual de cunoaşterea sistemelor de operare Windows şi a unora dintre componentele pachetului Microsoft Office.

Această carte este structurată pe 7 capitole.

Capitolul 1 începe cu o incursiune în istoria ştiinţei calculatoarelor. Tot în acest capitor sunt prezentate şi câteva dintre preocupările relative la generaţiile viitoare de calculatoare. Capitolul continuă cu o descriere a caracteristicilor principale ale unor tipuri reprezentative de calculatoare. Sunt prezentate, apoi, elementele principale ale calculatoarelor numerice: unitatea centrală de prelucrare, memoria, discurile magnetice şi optice, dispozitivele periferice de intrare şi dispozitivele periferice de ieşire. Tot în cadrul acestui capitol sunt introduse unele cunoştinţe generale referitoare la reţelele de calculatoare.

Capitolul 2 analizează problematica software-ului de sistem şi software-ului de aplicaţii al calculatoarelor. Sunt prezentate atât conceptele teoretice fundamentale ale sistemelor de operare cât şi evoluţia unor sisteme de operare semnificative. Tot în acest capitol se face o sinteză referitoare la limbajele de programare şi clasele de aplicaţii software.

Ca o exemplificare a conceptelor teoretice prezentate în Capitolul 2, Capitolul 3 este dedicat sistemelor de operare de tip Microsoft Windows, cu focalizare pe Windows 7. Sunt prezentate, în acest context, probleme referitoare la administrarea fişierelor şi a folderelor, lucrul cu aplicaţiile Windows, administrarea imaginilor şi secvenţelor video, configurarea calculatorului folosind Panoul de control, administrarea resurselor hardware şi software şi utilizarea resurselor Internet.

Capitolul 4 încearcă să facă o sinteză a elementelor care fac ca Microsoft Office să poată fi considerat un pachet integrat de aplicaţii. După o prezentare generală a elementelor şi a unor versiuni de pachete integrate Office, sunt exemplificate câteva elemente comune tuturor aplicaţiilor Office.

Capitolul 5 prezintă funcţionarea şi utilizarea editoarelor de documente, cu

exemplificare pe Microsoft Office Word 2007. Sunt prezentate funcţiile benzilor (panglicilor) cu comenzi ale programului Word 2007, modurile de vizualizare şi navigare în documente, modul în care se lucrează cu documentele şi operaţii ce se pot efectua asupra textului, cum se lucrează cu tabele, elemente de tip WordArt, şi cum se tipăresc documente.

În Capitolul 6 sunt prezentate funcţiile de bază ale unor programe folosite pentru prezentări, cu exemplificare pe Microsoft Office PowerPoint 2007. După prezentarea elementelor de bază, subcapitole distincte sunt dedicate operaţiilor efectuate asupra textului unei prezentări şi operaţiilor cu obiectele grafice. Capitolul se încheie cu explicarea modului în care se pot utiliza şabloanele de proiectare, scheme de culori diferite, diverse scheme de animaţie şi de tranziţie, şi modul în care se pot vizualiza prezentările realizate.

Ultimul capitol abordează problema calculului tabelar, și a posibilităților oferite de Microsoft Office Excel 2007 în acest domeniu. După ce sunt prezentate elementele și operațiile cu registrele de lucru și foile de lucru, un subcapitol separat este dedicat operațiilor la nivelul elementelor unei foi de lucru. Subcapitole extinse sunt dedicate formulelor și funcțiilor din Excel. Capitolul continuă cu utilizarea diagramelor și graficelor în Excel, iar în ultimul subcapitol este arătat modul cum pot fi utilizate informațiile din Excel ca o bază de date.

Autorul este conştient de faptul că orice carte, oricât ar fi ea de bine elaborată, nu poate răspunde tuturor întrebărilor dintr-un domeniu atât de vast. Pentru aceasta toate sugestiile sunt binevenite şi vor fi analizate în eventualitatea redactării unei noi ediții îmbunătățite.

1 ELEMENTE DE ARHITECTURA ŞI STRUCTURA CALCULATOARELOR

1.1 NOŢIUNI FUNDAMENTALE ÎN DOMENIUL CALCULATOARELOR

Dezvoltarea informaţională a omenirii poate fi considerată ca având cinci faze de "inovare" a informaţiei. Prima fază corespunde posibilităţilor "limbajelor vorbite"; a doua fază este legată de apariţia literelor şi dezvoltarea scrierii; a treia fază este determinată de apariţia tiparului şi dezvoltarea tipăririi, prevestind prima "explozie informaţională şi ştiinţifică" în interiorul naţiunilor şi depăşind frontierele dintre naţiuni. A patra fază constă în naşterea şi dezvoltarea telecomunicaţiilor, de la primul purtător de cod telegrafic şi până în zilele noastre, cu transmisii ale vocii, datelor, textelor şi imaginilor. A cincea fază - care a început odată cu calculatoarele electronice - este caracterizată prin posibilitatea de memorare (stocare), prelucrare şi transmitere a unei mari cantităţi de informaţie complexă, care nu mai poate fi manipulată prin mijloace convenţionale. Datorită acestui fapt a crescut în mod remarcabil câmpul de activităţi intelectuale ale omului şi se prevede, pentru viitorul apropiat, apariţia unei noi organizări sociale, bazată pe activitatea intelectuală şi progresul tehnic, denumită "societatea informaţională". În acest nou tip de societate producţia de informaţie va juca un rol mult mai important decât producţia industrială, iar calculatoarele şi telecomunicaţiile vor fi combinate sub formă de "sisteme de prelucrare a informaţiei". În literatura de specialitate este tot mai mult folosită denumirea de "epocă (eră) informatică" pentru a caracteriza această a cincea fază a dezvoltării informaţionale a omenirii [ŞERB11b].

Oamenii îşi prelucrează şi transmit informaţiile prin limbaje proprii. Aceste limbaje sunt însă prea complicate şi de neînţeles pentru calculatoare. Orice calculator numeric este capabil să rezolve probleme în folosul oamenilor prin executarea unei secvenţe de instrucţiuni numită **program**. Numărul de instrucţiuni pe care un calculator este capabil să le înţeleagă este însă unul limitat, şi orice program care ar trebui rulat pe un calculator trebuie tradus în acest limbaj înţeles de calculator. Acest limbaj se numeşte **limbaj maşină** şi toate programele care trebuie rulate pe un calculator trebuie transformate în instrucţiunile simple ale limbajului maşinii, pentru a putea fi recunoscute şi executate direct de către circuitele electronice ale unui calculator.

Pentru a reduce complexitatea şi costul componentelor utilizate la realizarea unui calculator, proiectanţii de calculatoare sunt obligaţi să conceapă setul de instrucţiuni primare (instrucţiunile maşină), înţelese de calculator, cât mai simplu cu putinţă. Aceste limbaje maşină extrem de simple sunt însă dificil de

utilizat de către oameni. Marea provocare a ştiinţei calculatoarelor, la începuturile sale, a fost acea de a identifica modul în care cerinţele utilizatorilor pot fi transpuse în activităţi capabile să le execute calculatoarele, rezultatul acestor activităţi fiind în final satisfacerea cerinţelor utilizatorului. Marea provocare a constat, aşadar, în a face posibil acest "dialog" între calculator şi oameni, şi de a găsi o punte de legătură între limbajul maşinii şi limbajul folosit de oameni.

Calculatoarele electronice de astăzi sunt maşini electronice care primesc comenzi pe baza unor semnale electrice. Semnalele cel mai uşor de înţeles sunt cele foarte simple, de tipul "închis" sau "deschis". Pe baza celor două astfel de semnale se poate crea un "alfabet" al calculatoarelor format numai din doua simboluri (litere) 0 şi 1. Aşa cum alfabetul unei limbi nu limitează numărul de cuvinte ce pot fi scrise cu aceste caractere, nici numărul de cuvinte create cu simbolurile 0 şi 1 nu este unul limitat. Cele două simboluri 0 şi 1 sunt considerate "alfabetul" pentru maşină, iar toate informaţiile cu care lucrează calculatoarele sunt numere scrise în baza 2, sau **numere binare**. În acest fel, fiecare "literă" din limbajul calculatoarelor devine o "cifră binară" sau un "bit". Aceşti biţi sunt combinaţi împreună pentru a codifica instrucţiunile şi datele cu care poate lucra un calculator.

Ideea de a comunica cu calculatoarele prin intermediul numerelor de tip binar este una veche în ştiinţa calculatoarelor. Scrierea de comenzi pentru calculatoare folosind doar simbolurile 0 şi 1 este însă una foarte dificilă şi obositoare. De aceea pionierii ştiinţei calculatoarelor au preferat să inventeze nişte programe care să translateze în binar comenzile pentru calculator, comenzi pe care ei le concepeau într-un limbajul simbolic. Aceste programe au fost numite **asambloare**, şi, deci, rolul asambloarelor era acela de a translata versiunile simbolice ale instrucţiunilor în versiuni binare ale acestor instrucţiuni. Numele folosit pentru limbajele simbolice este acela de "**limbaj de asamblare**".

S-ar putea considera, pentru scop didactic, că limbajul utilizat de oameni ar fi limbajul X, iar limbajul maşinii ar fi limbajul Z. Pentru a putea executa un program scris în limbajul X, pe un calculator capabil să înţeleagă instrucţiuni în limbajul Z, ar trebui să existe o metodă prin care toate instrucţiunile din limbajul X să poată fi „traduse" în limbajul Z. Un instrument capabil de a înlocui fiecare instrucţiune din limbajul X cu o secvenţă echivalentă de instrucţiuni în limbajul Z, şi de a executa apoi această secvenţă de instrucţiuni în limbajul Z, se numeşte **interpretor**. Un **translator** este capabil să traducă fiecare instrucţiune dintr-un limbaj X într-o secvenţă de instrucţiuni în limbajul Z, şi să genereze un nou program, pe care apoi să-l execute. Diferenţa dintre un interpretor şi un translator este, deci, aceea că în timp ce un interpretor examinează pe rând fiecare instrucţiune din limbajul X, o transformă în secvenţe de instrucţiuni echivalente în limbajul Z, o execută şi apoi trece la o nouă instrucţiune a programului, s.a.m.d., un translator transformă, în întregime, instrucţiunile din limbajul X în instrucţiuni

în limbajul Z, generează un nou program pe care îl execută, ignorând, în faza de execuţie, programul scris în limbajul X.

Într-o oarecare măsură, traducerea şi interpretarea sunt similare, dar pentru ca ele să poată fi şi aplicabile în practică ar trebui ca cele două limbaje X şi Z să nu fie prea diferite. Dacă ar fi avute însă în vedere limbajele utilizate uzual de oameni şi limbaje executate de maşini, diferenţele dintre ele sunt însă, în mod evident, foarte mari. De aceea este corectă ideea de a crea un alt set de instrucţiuni, un limbajul Y, care să fie mai convenabil pentru programatori şi mai puţin orientat spre limbajul maşină. În acest fel algoritmii utilizatorului, gândiţi în limbajul X, ar putea fi traduşi mai uşor în limbaj Y, iar programele create astfel în limbajul Y ar putea fi traduse mai uşor în limbaj Z. În acest fel fiecare limbaj poate folosi predecesorul său ca bază. În loc de a gândi problema în termeni de limbaje, ea poate fi gândită la fel de bine în termeni de **maşini virtuale** capabile să lucreze cu instrucţiuni în limbajele X, Y, Z, etc. Folosind acestă tehnică orice sistem de calcul poate fi văzut ca o succesiune de straturi sau nivele, sau ca o succesiune de maşini virtuale, unul deasupra celuilalt, nivelul de jos fiind cel mai simplu, iar nivelul de sus cel mai complicat.

1.2 MAŞINI VIRTUALE MULTI-NIVEL

Privind din perspectiva maşinilor de calcul organizate pe mai multe nivele, cel mai simplu mod de a privi un sistem de calcul este acela în care un sistem de calcul poate fi văzut ca o maşină cu trei nivele. Pe nivelul cel mai de jos este hardware-ul, iar pe cel mai de sus este software-ul, nivelul intermediar fiind constituit din setul de instrucţiuni al calculatorului (Fig. 1.1).

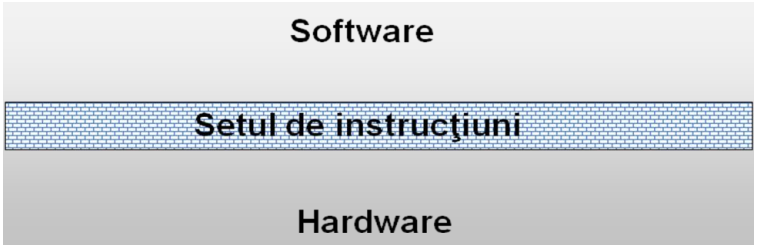

Fig. 1.1 Calculator cu trei nivele

Analizând calculatoarele, în modul cel mai general de abstractizare, nivelul de bază a oricărui calculator este, aşadar, **hardware**-ul calculatorului. Hardware-ul calculatoarelor actuale este format din circuite electronice, microprocesoare, memorii, dispozitive de intrare/ieşire, etc., deci din obiecte fizice, tangibile, de tipul circuitelor integrate, memoriilor, plăcilor cablate, surselor de alimentare, monitoarelor, imprimantelor, etc. Hardware-ul, ar

reprezenta deci ansamblul componentelor fizice şi tehnice cu ajutorul cărora datele şi programele se pot introduce, stoca, prelucra, livra, verifica, transmite, etc. Hardware este, deci, termenul general care desemnează resursele fizice (circuitele, dispozitivele şi echipamentele componente) ale unui calculator numeric.

Nivelul superior este constituit din **software**, care constă din programe scrise pe baza unor algoritmi. Software reprezintă ansamblul componentelor logice de tipul programelor, procedurilor, rutinelor, modulelor, etc., care dirijează funcţionarea componentelor hard şi asigură prelucrarea datelor, cu scopul de a rezolva diverse probleme concrete. Există două tipuri de software: software de sistem şi software de aplicaţii. Cea mai importantă componentă a software-ului este **sistemul de operare**, cel care administrează şi controlează hardware-ul şi furnizează mediul de operare pentru toate programele. **Software-ul de aplicaţii** este numele dat tuturor programelor care sunt destinate prelucrărilor dorite de utilizatorii de calculatoare.

Nivelul care face posibil dialogul hardware-ului cu software-ul este **setul de instrucţiuni** (limbajul maşină) pe care este capabil să-l execute un calculator. Pentru a fi posibil ca software-ul să poată fi executat pe hardware-ul unui calculator, aceste soft trebuie tradus în limbajul specific setului de instrucţiuni al acelui calculator, aceste instrucţiuni fiind cele care activează componentele hardware, în conformitate cu programele în lucru.

Programele scrise în limbaj de asamblare îi cer însă programatorului să scrie o linie de program pentru fiecare instrucţiune pe care o va executa calculatorul, şi astfel programatorul este obligat să gândească asemenea maşinii de calcul pe care ar rula programele sale. Evident, acest lucru este neconvenabil. Pornind de la observaţia că orice program scris în limbaj de asamblare poate fi tradus în instrucţiuni binare prin intermediul unui **asamblor**, cercetătorii din ştiinţa calculatoarelor au ajuns la concluzia că nimic nu îi împiedică să creeze un program care să efectueze o translatare dintr-un limbaj de nivel superior în limbaj de asamblare, translatarea în instrucţiuni maşină putând fi efectuată ulterior, fără nici un fel de probleme, prin intermediul unui asamblor. Astfel, a apărut, şi s-a dezvoltat, ideea **limbajelor de nivel înalt (limbajelor de programare)** şi a **translatoarelor** (**compilatoarelor**) care să traducă aceste programe în limbaj de asamblare.

Avantajele limbajelor de nivel înalt sunt foarte mari. În primul rând, programatorul îşi poate gândi algoritmul programului său într-un limbaj mai natural, folosind cuvinte din dicţionar si notaţii algebrice, iar programul pe care îl realizează seamănă mai mult cu un text decât cu o sumă de cifre binare. Un al doilea avantaj este acela că elaborarea programelor necesită mai puţin timp, limbajele de nivel înalt fiind mai concise şi necesitând mai puţine instrucţiuni pentru a exprima o ideea. Un alt avantaj este acela ca programele sunt

independente de calculatoarele pe care ele urmează a fi rulate, deoarece compilatoarele şi asambloarele vor translata programele din limbajele de nivel înalt în instrucţiunile binare ale fiecărui calculator. Nu în ultimul rând, trebuie avut în vedere şi faptul că limbajele de programare sunt oarecum create în concordanţă cu scopul utilizării lor. Astfel, limbajul Fortran a fost creat pentru calcule ştiinţifice, limbajele Cobol, Access, etc. pentru prelucrarea datelor, limbajul Lisp pentru manipularea simbolurilor, şi aşa mai departe.

Cele mai multe calculatoare pot fi însă analizate din perspectiva a mai mult decât cele trei nivele menționate anterior. O astfel de modalitate de prezentare este cea pe 7 nivele și care este prezentată în figura 1.2.

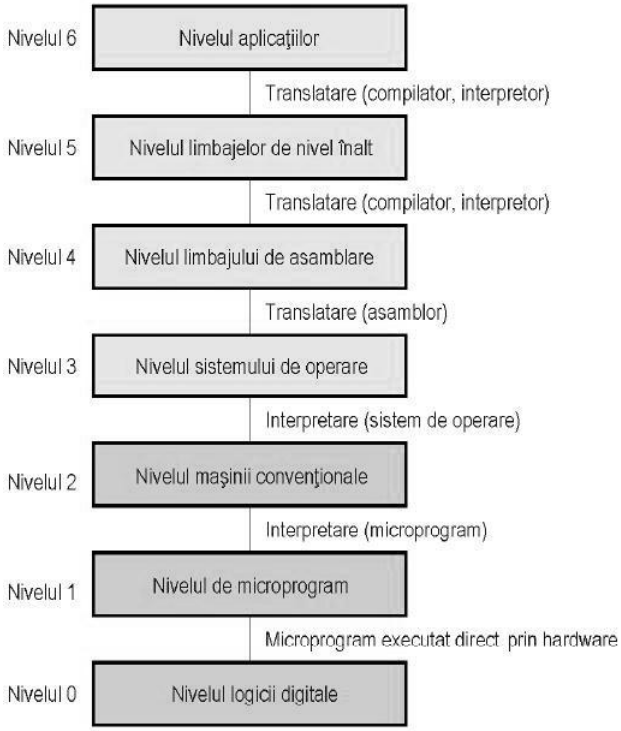

Fig. 1.2 Calculator cu şapte nivele

Nivelul 0, numit *nivelul logicii digitale,* este reprezentat de componentele hardware ale calculatorului (maşina fizică). Circuitele acestui nivel execută instrucţiunile maşină ale nivelului 1. Elementele de bază ale acestor circuite sunt *porţile logice*, fiecare poartă fiind formată la rândul ei dintr-un număr de tranzistoare. O poartă logică are una sau mai multe intrări digitale (semnale reprezentând 0 logic sau 1 logic), şi are ca ieşire o funcţie simplă a acestor intrări, de exemplu ŞI logic, SAU logic.

Nivelul 1, numit *nivelul de microprogram*, interpretează instrucţiunile nivelului 2, pentru fiecare instrucţiune a acestui nivel existând câte un microprogram.

Fiecare microprogram defineşte în mod implicit un limbaj de nivel 2 şi o maşină virtuală. De obicei există multe similarităţi între maşinile virtuale de nivel 2, chiar şi în cazul calculatoarelor diferiţilor producători. Acest nivel se numeşte *nivelul maşinii convenţionale*. Atunci când se descrie setul de instrucţiuni al unui calculator, se descrie de fapt maşina virtuală de nivel 2, şi nu maşina reală de nivel 1. Sunt descrise deci instrucţiunile interpretate de către microprogram, şi nu instrucţiunile executate direct prin hardware.

Nivelul 3 este de obicei un nivel hibrid, deoarece multe din instrucţiunile limbajului său sunt prezente în cadrul instrucţiunilor nivelului 2. Există în plus un set de noi instrucţiuni, o organizare diferită a memoriei, posibilitatea de execuţie a mai multor programe în paralel şi alte facilităţi. Noile facilităţi adăugate la nivelul 3 sunt realizate cu ajutorul unui interpretor, numit *sistem de operare*, iar instrucţiunile identice cu cele ale nivelului 2 sunt executate direct prin microprogram. Nivelul 3 este numit *nivelul sistemului de operare*. Un sistem de operare reprezintă un ansamblu de programe care asigură exploatarea optimă a resurselor hardware şi software ale unui sistem de calcul. Sistemele de operare au fost create pentru simplificarea activităţii de programare, utilizarea optimă a posibilităţilor de lucru ale echipamentelor şi reducerea intervenţiei utilizatorilor în cursul execuţiei programelor.

Nivelele 0-3 nu sunt destinate utilizării directe de către programatorii obişnuiţi, ci pentru rularea translatoarelor şi interpretoarelor scrise de programatorii de sistem. Nivelul 4 şi nivelele superioare sunt destinate programatorilor de aplicaţii.

Nivelul 4 este *nivelul limbajului de asamblare*. Programele scrise în limbaj de asamblare sunt translatate în limbajul nivelului 1, 2 sau 3 şi apoi interpretate de către maşina virtuală corespunzătoare.

Nivelul 5 constă din limbajele destinate programatorilor de aplicaţie, fiind numit *nivelul limbajelor de nivel înalt*. Programele scrise în aceste limbaje sunt translatate în limbajele nivelului 3 sau 4 cu ajutorul compilatoarelor sau interpretoarelor.

Nivelul 6 reprezintă *nivelul aplicaţiilor*. Constă din colecţii de programe destinate unor domenii specializate, de exemplu pentru administraţie, economie, proiectare asistată de calculator, grafică etc.

Fiecare nivel reprezintă o abstractizare distinctă, cu diferite obiecte şi operaţii. Setul tipurilor de date, a operaţiilor şi facilităţilor fiecărui nivel reprezintă arhitectura nivelului respectiv.