A projekt az Európai Unió társfinanszírozásával, az Európa terv keretében valósul meg.

OPERÁCIÓS RENDSZEREK

HEFOP 3.3.1–P.-2004-06-0071/1.0

Ez a kiadvány a
„Gyakorlatorientált képzési rendszerek kialakítása és minőségi fejlesztése az agrár-felsőoktatásban”
című program keretében készült

OPERÁCIÓS RENDSZEREK

Szerkesztő:

Dr. Harnos Zsolt
Budapesti Corvinus Egyetem

Dr. Herdon Miklós
Debreceni Egyetem

Szerző:

Dr. Berke József
Pannon Egyetem

Csák Máté
Pannon Egyetem

Lektor:

Dezső Ottó
Szent István Egyetem

Nagy Sándor
Pannon Egyetem

ISBN 978-963-9732-54-4

E tankönyv teljes mértékben megegyezik a Debreceni Egyetem honlapján, a http://odin.agr.unideb.hu/hefop/ elérési úton megtalálható, azonos című tankönyvvel.

Első kiadás

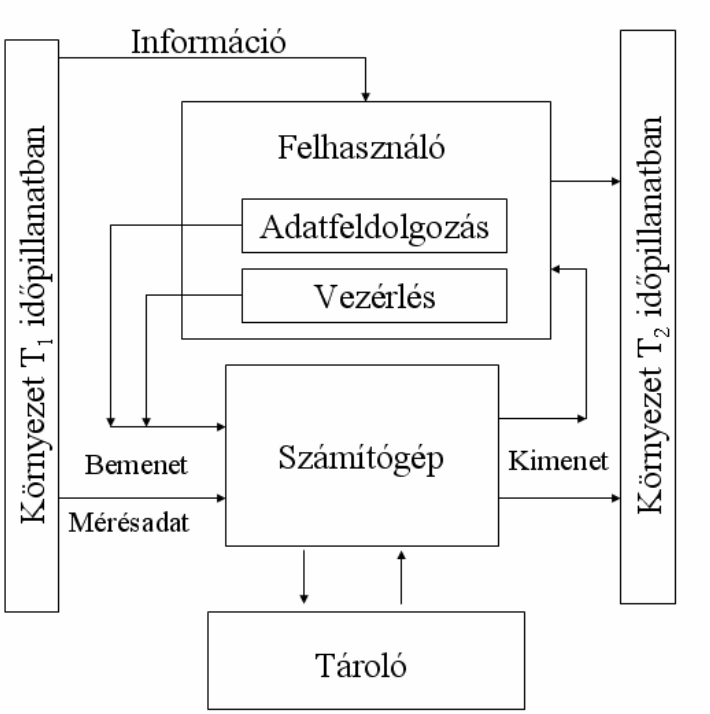
Kiadó:
Debreceni Egyetem Agrár- és Műszaki Tudományok Centruma
Agrárgazdasági és Vidékfejlesztési Kar

Debrecen, 2007.

Tartalomjegyzék

Bevezetés

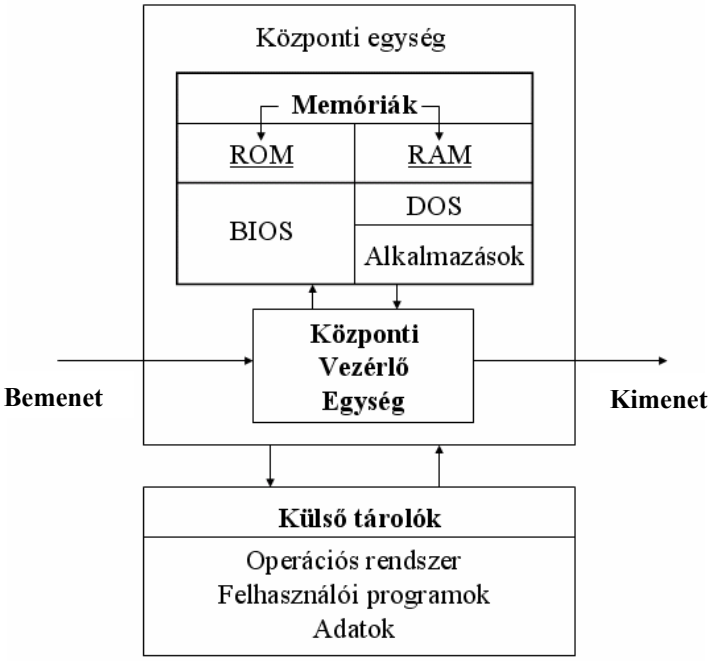
Többféle dolog is eszünkbe juthatna, ha halljuk ezeket a szavakat, „operációs rendszerek”, eszünkbe juthatna például szavanként értelmezve, munka és szervezettség. Mi is lenne, ha ezt tovább gondolnánk? Könnyen eljuthatnánk a „munkaszervezési módszerek” kifejezésig. Mégsem ezt tesszük. Vajon miért van az, hogy, ha halljuk, operációs rendszerek, egy számítógépünk használatához nélkülözhetetlen programra, esetleg egy cégre gondolunk. Mert a mindennapokban egyszerűsítünk, használjuk dolgainkat, s észre se vesszük részei vagyunk környezetünk változásának, de nagy különbség van, hogy ezt úgy tesszük, hogy egyszer sem gondoltuk végig, hogy ez hogyan is történik, vagy egyszer már végig gondoltuk, csak éppen most ez nem fontos hogy eszünkbe jusson. Persze az is előfordulhat, hogy mégis végig kell gondolnunk. Milyen operációs rendszert is vegyek, vagy egy adott esetben milyen operációs rendszert is ajánljak. Ilyenkor miből indulhatunk ki? Célszerű a „munkaszervezési módszerek” kifejezésből.



B-1. ábra: A személyi számítógép mint információ transzfer

A szervezett folyamatok, mint minden folyamat anyag, energia és információ áramból, és ezek keletkeztetőiből, átalakítóiból és elfogadóiból tevődik össze. Azt tudjuk, hogy a számítógép egyik nézetében információ-átalakító *(B-1. ábra)*. Ezt figyelembe véve az operációs rendszert csak mint egy nagyobb folyamat részeként, pontosabban kiszolgálójaként tekinthetjük, ezért választani tudásunk csak az információs rendszerek további ismerete esetén lehet teljes. Tudásunk csak apró lépésekben bővíthető, ezen tankönyv csak egy lépés a számítógép alkatrészek halmazának, elemeinek és működésüknek ismeretétől a vállalati, és a vállalati rendszereket is magába foglaló informatikai rendszerek megismerésében.

A számítógép egy másik nézetében alkatrészekből összeállított gép, ahogy a számítógépes architektúrák tantárgyból is megtudhatták. *(B-2. ábra)*



B-2. ábra: A személyi számítógép mint gép

Megismerhették, hogy az alkatrészek csak mikroprogramjaik segítségével tudnak együttműködni a teljes számítógéppé válás érdekében, és most felismerhetik, hogy a felhasználó okozta környezetváltozás érdekében alkalmazott programok csak egy köztes program, egy virtuális gép[1], az operációs rendszer segítségével férhetnek hozzá az alkatrészekhez, mint erőforrásokhoz.

Tehát mondhatjuk, ha ezentúl az „operációs rendszer” szavakat halljuk, és eszünkbe jut a „munkaszervezési módszerek” kifejezés, két dologra kell, hogy gondoljunk, egyrészt, az adott számítógép egy szervezet kiszolgálójaként betölt egy szerepet, másrészt, szerencsés esetben ehhez a szerephez egyfajta optimalizációs folyamat eredményeként hozzá lett rendelve egy funkcióját betölteni képes alkatrészhalmaz, melyet csak a megfelelően kiválasztott operációs rendszer tud célszerűen munkára szervezni. Könyvünk ebben, a megfelelő kiválasztásban próbál segítséget nyújtani. Az első fejezetben történeti és elméleti ismereteket nyújtunk, a következő fejezetben két lehetséges, karakterében erősen különböző operációs rendszert ismertetünk, majd a harmadik fejezetben, az előzőekre hivatkozva, ill. kiegészítve, döntési szempontokat járunk körbe.

Ismereteik bővítése érdekében forgassák haszonnal:

A szerzők

[1] virtuális gép ~ kiterjesztett gép, Tanenbaum [1999], 23. o.

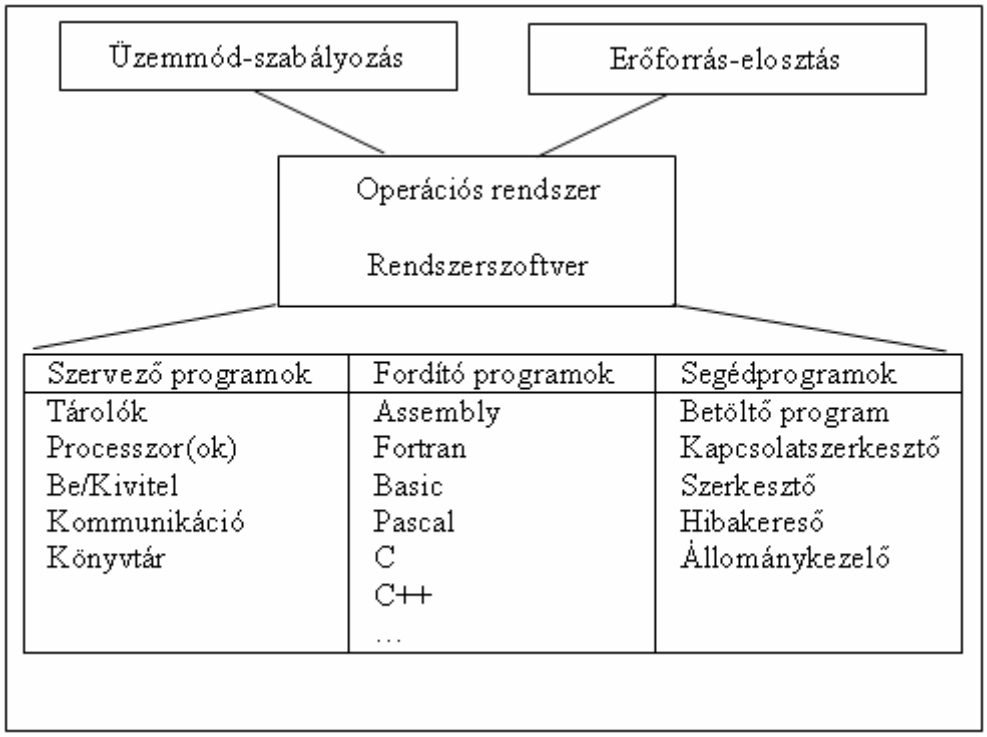
1. OPERÁCIÓS RENDSZEREK ELMÉLETI ALAPJAI

1.1. Operációs rendszerek általános jellemzői

1.1.1. Operációs rendszerek kialakulása

Az első digitális számítógép még mechanikus volt (Charles Babbage 1792-1871), igaz megfelelő alkatrészek hiányában nem működött megfelelően. Mechanikus kiépítése miatt szabad programozásról, ehhez szükséges operációs rendszerről még nem beszélhetünk.

Az első Neumann elvű számítógépek – **első generáció** (1945-1955), vákuumcsövek – még huzalozás cserével, kapcsolótáblákkal, az időszak vége felé lyukkártyákkal oldották meg a programok cseréjét, mely programok még nagyrészt numerikus számítások voltak.



1-1. ábra: Az operációs rendszer szolgáltatásai[2]

A **második generáció** (1955-1965) – tranzisztoros számítógépek jellemzője a korai kötegelt parancsfeldolgozás volt, tudományos, mérnöki számítások céljából.

Ez a megoldás a következőket jelentette[3]:

A beviendő adatokat és a programokat lyukkártyákon rögzítették, a lyukkártyák adatait mágnesszalagra másolták, egy kisebb teljesítményű számítógépen (IBM 1401), egy szalagra akár több programot is.

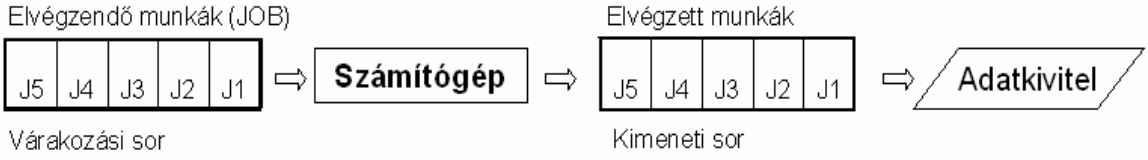
A nagyobb számítási teljesítményű számítógép a feladatait erről a szalagról olvasta be, az eredményeket mágnesszalagra írta (IBM 7094).

Az eredményeket tartalmazó szalag tartalmát egy kisebb teljesítményű számítógép nyomtatta (IBM 1401) (*1-2. ábra*).

Ekkor merült fel a fordítóprogramokkal egybeépült, különféle perifériákat kezelni tudó program szükségessége, mely a szalagokról betöltött alkalmazások forráskódját fordítani,

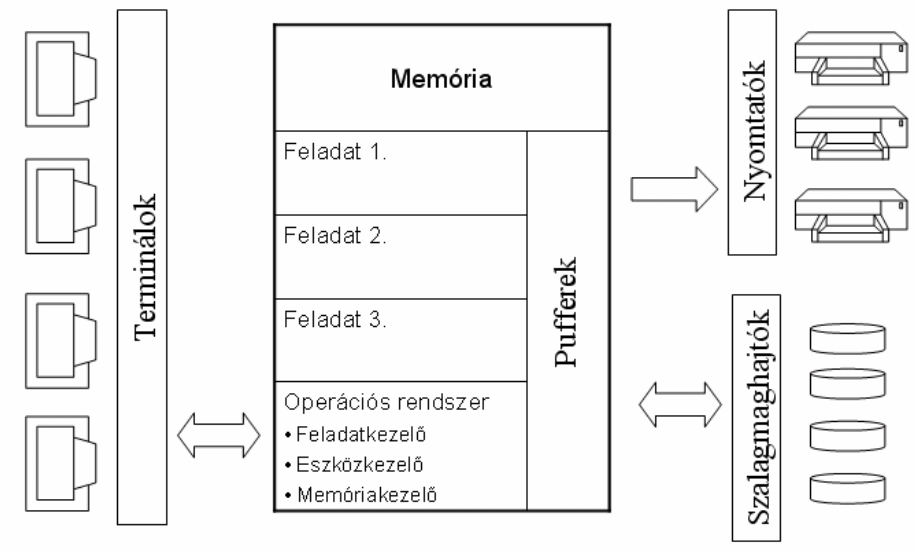
[2] Informatika [1995], 92. o.
[3] Tanenbaum [1999], 25. o.

futtatni tudta, az eredményeket pedig az aktuális kimenetre irányíthatta, például FMS – Fortran Monitor System, vagy IBSYS az IBM 7094-en. A programok két félék lettek, lett egy program mely a számítógépet működtette, később ebből lett az operációs rendszer, és ezidőtől külön beszélhetünk programokról, melyek később az alkalmazások nevet kapták.



1-2. ábra: Korai kötegelt üzemmód

A **harmadik generáció** (1965-1980) - integrált áramkörökkel szerelt számítógépek jellemzője a multiprogramozás. Az integrált áramköröknek köszönhetően létrejöhetett egy számában és fajtájában tetszőleges perifériával, de azonos operációs rendszerrel (OS/360) ellátott számítógépcsalád az IBM ötlete alapján. A lassú perifériák miatt a megnőtt teljesítményű processzor a kötegelt feldolgozás során, az adatmozgatás idején munka nélkül maradt, a nagy mennyiségű adatbeviteli és adatkiviteli igényekkel fellépő üzleti alkalmazásoknál a processzor kihasználtsága 10-20%-ra csökkent. A megoldást a memória növelése és partícionálása, azaz részekre bontása jelentette. Ha külön-külön rendelhetünk feladatot minden egyes memória partícióhoz a következő előnyhöz jutunk, amikor egy feladat be- és kiviteli (az angol „Input" (bevitel) és „Output" (kivitel) kifejezések elterjedt rövidítéssel I/O, magyar irodalomban gyakran alkalmazott fordításával B/K) művelethez ér, a processzor egy mások feladaton dolgozhat. Igaz, mint minden problémamegoldás, ez is új problémát szült, kiszámíthatatlanná vált egy feladat végrehajtási ideje a felváltva végrehajtott feladatok nagy száma miatt, hisz egy feladat végrehajtása után az üresen maradt partícióba máris egy másik feladat kerülhetett betöltésre. Azt a technikát mely lehetővé teszi a több B/K csatornáról érkező feladatoknak hogy egy ideiglenes memória területen várakozzanak, míg felszabadul egy memóriapartíció spooling technikának (Simultaneous Peripheral Operation On Line) nevezzük. A kiszámíthatatlan feladat-végrehajtási idő kikényszerítette az „időosztás" technikájának a megvalósítását. Ez a technika lehetővé teszi az operációs rendszer részére a gyors, ciklikus, azonos időközönkénti feladatváltást. A korszak végére a memória partícionálás, a spooling technika, és az időosztás közös használata már lehetővé tette az interaktív terminálok használatát, kialakult az első felhasználói felület. *(1-3. ábra)*



1-3. ábra: Terminálokat kiszolgáló (mainframe) számítógép memóriafelosztása

A korszak jellemző operációs rendszerei az OS/360-on kívül a Multics, MVS. Megjelentek az első miniszámítógépek PDP nevű sorozatként, valamint felbukkant a PDP-7 jelű számítógépre az első egyfelhasználós operációs rendszer, azaz a Multics-ból megszületett az első Unix.

A **negyedik generáció** (1980-tól napjainkig) az LSI (Large Scale Integration – magas integráltságú) áramkörök fejlődésének köszönheti meglétét, ennek a korszaknak jellemző eszköze a személyi számítógép (Personal Computer), melynek architektúrája a PDP-11 miniszámítógépével nagyjából megegyezik. Közös jellemzőjük az alkatrészeikhez illesztett operációs rendszer. Keletkezésük hajtómotorja a kis válaszidejű interaktivitás volt.

Mielőtt a PC-k karrierje megindult volna, az olcsó, tömegigényt kielégítő otthoni számítógépek (home computer) váltak széles körben elérhetővé. Ezek az operációs rendszer nélküli 8 bites, egy felhasználós gépek a csak olvasható memóriájukban (ROM) hordozták a korai kötegelt parancsfeldolgozással megegyező programbetöltő, programfuttató (Basic interpreter) képességüket, ez a mágnesszalag-kazetták, floppy lemezek és az ugrálós basic világa volt. Megjelenítőként televízió készülék szolgált, esetleg nyomtató. Szövegszerkesztők, játékok, kisebb üzleti alkalmazások futtatására voltak alkalmasak[4]. Manapság programozható menedzser-kalkulátorokként, „okostelefonként" élnek tovább, időlegesen, hisz az operációs rendszerek egy része elkezdett átköltözni az okostelefonokba is. Az otthoni számítógépeket jellemző márkanevek: Apple, Atari, Commodor, IBM.

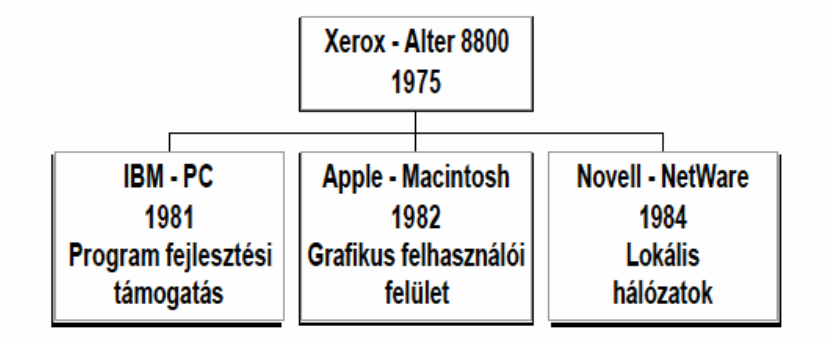
1-4. ábra: IBM PC[5] 1981.

(Intel 4.77MHz, 8088 processzor, két 5.25" 160K hajlékony lemez meghajtó, Basic Interpreter a ROM-ban, a lemezről betölthető operációs rendszer: DOS 1.0.)

A személyi számítógépek korai operációs rendszereinek nagy része karakteres felhasználói felülettel (UI) rendelkezett, klasszikus szerkezetű *(1-4. ábra)*, valamint egy felhasználós volt, egyszerre csak egy alkalmazást tudott futtatni, például CP/M, IBM-DOS,

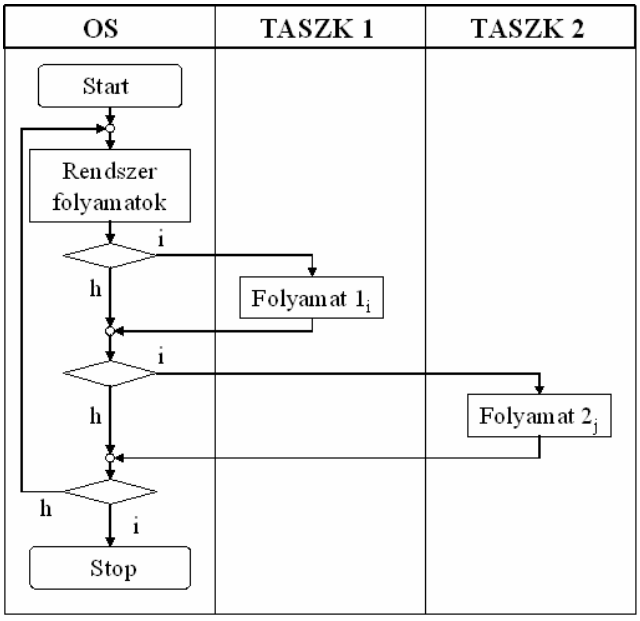
[4] Rajongói oldalakról az otthoni számítógépek PC-n futó emulátorai, programjai nagyrészt letölthetőek
[5] http://www.columbia.edu/acis/history/

MS-DOS. 1982-ben jelent meg első grafikus felhasználói felülettel (GUI) rendelkező személyi számítógép Macintosh néven.



1-5. ábra: A személyi számítógép születése[6]

A 8 ill. 16 bites korszak lezárásaként a PC-ken is megjelentek a 32 bites, grafikus felhasználói felületű, többfeladatos operációsrendszerek *(1-6. ábra)*, 1989-ben az Apple System 7 Macintosh rendszere, 1992-ben az Unix szabadterjesztésű klónjai az első Linux disztribúciók, 1994-ben az IBM OS/2 Warp 3 rendszere[7], 1995-ben pedig a Microsoft Windows 95 rendszere.



1-6. ábra: Többfeladatos operációs rendszer működési vázlata

Mielőtt rátérnénk napjaink operációs rendszereire essék szó a **klasszikus hálózati operációs** rendszerekről is. Nyilvánvaló, hogy a mainframe és a miniszámítógépek operációs rendszerei a hálózati csatolók beszerelése után, melyek csak újabb B/K csatornákat nyitnak, erőforrásaik megosztására alkalmasak, igy a kliens-szerver típusú hálózatokban a szerver oldalon nagyszerűen működnek (VMS, Unix, Solaris, Novell[8]), a személyi számítógépek pedig intelligens terminálként a kliens oldalon tölthetik be szerepüket (MS Windows, eComSation, Linux, MacOs). A személyi számítógépek teljesítmény növekedése pedig

[6] Knapp [1998], 32. o.
[7] Az IBM Warp 4 operációs rendszere eComStation néven él tovább - http://www.ecomstation.com/
[8] Fejlődése 1981-ben, a CP/M rendszerből kezdődött, a fejlesztés célja a keménylemezmegosztás volt

lehetővé teszi SOHO (Small office/Home office) környezetben a peer-to-peer hálózatokon keresztüli kölcsönös erőforrás megosztást, persze megfelelő illesztő programok használatával.

Napjainkra az átviteli szabályok alapján is osztályozhatjuk a hálózatokat TCP/IP (Unix), IPX (Novell), Netbeui (MS LAN, régebben IBM LAN), azonban a zárójelben található fejlesztői környezet szerinti azonosítás már idejét múlt, a megfelelő segéd és beépített programok miatt már mindenféle operációs rendszer szinte mindenféle hálózathoz csatlakozhat, ill. hálózatot kiszolgálhat. Például Unix/Linux – MS LAN interfész a Samba segédprogram, a Novell újabb változatai futnak TCP/IP felett.

Az eddigi történeti áttekintés során főként abból indultunk ki, hogy a számítógép kapacitása többszörösen meghaladja egy-egy feladat megoldásához szükséges kapacitást, pedig már korai időktől kezdve előfordultak olyan extrém számítási igényű feladatok, ahol a számítás szinte már nem végezhető el a véges időben. Ilyen esetekben nincs más választás, mint a **feladatosztás**. A feladatosztás két lehetséges megoldása közül az egyik, a szorosan kapcsolt rendszer *(1-7. ábra)*, amikor egy buszra igyekszünk a lehetséges legtöbb processzort felfűzni, ebben az esetben a meglevő mainframe kiszolgáló operációs rendszert pl. UNIX-ot fejlesztenek tovább.

System	Site	Number of procs	Proc speed (Mflops)	Pk speed (Gflops)	Tot mem (GB)	Available (%)
Primary production systems						
IBM SP (Power3-II): Blue Horizon	SDSC	1,152	1,500	1,728	576	100
IBM p690 HPCs	U Texas	4x16	5,200	333	128	50
IBM SP (P2SC)	U Michigan	176	640	113	96	75
IBM SP (Power3-II)	U Michigan	24	1,500	36	24	100
Cray T3E	U Texas	272	600	163	37	75
Cray SV1	U Texas	16	1,200	19	16	10
Alternate architecture systems						
AMD Athlon cluster	U Michigan	100->256	3,200	320->819	50->128	100
HP V2500s	Caltech	2x64	1,760	2x113	2x64	50
Sun HPC 10000	SDSC	64	800	51	64	25

1-7. ábra: Szuperszámítógépek 2002-ben

(A táblázatban szereplő FLOPS értékek a processzoroknál (elsősorban) nagygépes környezetben alkalmazott sebesség-mérték: „FLoating-point Operations Per Second”, azaz másodpercenként elvégzett lebegőpontos műveletek száma.)

A másik lehetőség a lazán kapcsolt rendszer, amikor már egy meglevő hálózat elemeit kapcsoljuk össze operációs rendszertől függő segédprogramokkal, például SETI projekt[9], ahol a csatlakozott számítógépek felhasználójukat nem zavarva töltik le részfeladatukat, végzik el a rájuk osztott műveleteket, töltik fel eredményeiket végtelennek tűnő ciklusban idegen civilizációk nyomát keresve a digitalizált rádióhullámokban.

Napjainkhoz érve még egy speciális hálózati lehetőségre kell figyelmet fordítanunk, ez, a ma vékony kliensnek nevezett eszköz, mely a régi terminálok továbbélését teszi lehetővé, például IBM NetPC, mely terminál az Interneten keresztül csatlakozhat Windows, Unix szerverekhez, esetleg IBM mainframekhez. Ez előre vetít egy lehetséges világot, hogy nemcsak adatainkat tároljuk mondjuk az Internet szolgáltatónknál, hanem még a szövegszerkesztőnket is az Internet szolgáltató mainframjén futtatjuk.

[9] http://www.seti.org/

Elérkezve napjainkhoz, a mindennapok munkájához, otthoni számítógépeinkhez láthatjuk, hogy a személyi számítógépek világában élünk, ezeket használjuk. Megfigyelhetjük, hogy operációs rendszereik szinte a következőkre szorítkoznak MS Windows, Linux disztribúciók, MacOs. Magyarországon az irodákban, üzleti életben, kormányzati hivatalokban úgy tűnik a Microsoft az egyeduralkodó, de halljuk Hamburgban, Párizsban a közhivatalok már áttértek a Novell/SuSe Linux disztribúcióra…

1.1.2. Az operációs rendszerek csoportosítása

A hagyományos megközelítést (mely szerint az operációs rendszerek a számítógépek működéséért felelős szoftverek) általánosítva (talán egy kis túlzással) azt mondhatjuk, hogy minden ICT[10] eszközben egy-egy operációs rendszer rejtőzik: a számítógépben, a mobiltelefonban, a gépkocsiban, de még az otthonunkban megtalálható elektronikus eszközök jelentős részében is (és az előrejelzések szerint az intelligenciával rendelkező eszközök száma a jövőben várhatóan csak nőni fog). Ha most elvonatkoztatunk a beágyazott rendszerektől (amelyek valamely célhardver vezérléséért felelősek), és vizsgálódásunkat kizárólag a számítógépes rendszerekre szűkítjük, akkor is azt kell látnunk, hogy a jelenleg ismert, alkalmazott (vagy éppenséggel a mára idejét múlt) operációs rendszerek száma százas nagyságrendű. Az természetes, hogy minden egyes operációs rendszer többé-kevésbé eltér más rendszerektől, az azonban már talán nem az, mindezen sokszínűség ellenére is akadnak olyan közös jellemzők, amelyek segítségével az operációs rendszereket kategorizálhatjuk. Hangsúlyozni szeretnénk azonban, hogy az itt közölt csoportosítás nem feltétlen teljes (további szempontok elképzelhetőek), valamint azt, hogy az egyes rendszerek besorolása egy-egy csoportba (az esetek többségében) nem kizárólagos.

Operációs rendszer-kategóriák (példák)

1) Felhasználói felület szerint:
   a) karakteres (CP/M, DOS)
   b) grafikus (Apple Mac OS)
2) Felhasználók száma szerint
   a) egy-felhasználós (BeOS)
   b) több-felhasználós (Microsoft Windows XP)
      i) hálózati (Novell Netware)
3) Folyamatkezelés szerint
   a) kötegelt (Microsoft MS DOS)
   b) multiprogramozott
      i) valós idejű (BeOS, QNX)
      ii) időosztásos (Multics, UNIX)
4) Hardver-architektúrák szerint
   a) számítógép-ketegóriák
      i) szuperszámítógépek (UNIX)
      ii) nagy számítógép („mainframe”) (SUN Solaris)
      iii) miniszámítógép (IBM OS/400)
      iv) személyi számítógép:
         (1) szerver (Microsoft Windows 2000, Linux),
         (2) munkaállomás (IBM OS/2 Warp)
      v) mikroszámítógép (Commodore 64)

[10] ICT: (Information and Communication Technology) info-kommunikációs technológia. Manapság a számítástechnika helyett, annak kiterjesztett értelmezésében (beleértve nem csak a hagyományos számítógépeket, hanem a kommunikációs eszközöket és egyéb „intelligens” berendezéseket is) használt fogalom.

vi) kézi számítógép (PalmOS)
   b) processzor-architektúrák szerint
      i) CISC alapú (Linux, Microsoft Windows)
      ii) RISC alapú (Hewlett-Packard HP-UX)
   c) sínrendszer alapján
      i) 16 bites (IBM PC DOS)
      ii) 32 bites (Microsoft Windows XP)
      iii) 64 bites (Macintosh OS X)
5) Jogállás szerint
   a) szerzői jogvédelem alá tartozó (SCO OpenServer)
   b) nyílt forráskódú (Linux)
6) „Történelmi" kategóriák
   a) korai operációs rendszerek
   b) UNIX-alapú rendszerek
      i) UNIX verziók, POSIX-kompatibilis rendszerek
      ii) Linux disztribúciók
   c) Windows rendszerek

Az operációs rendszerek csoportosításának legtriviálisabb, ugyanakkor legkevésbé egyértelmű szempontokat meghatározó módszere lehet a felhasználói preferenciák szerint történő csoportosítás. Ide sorolhatók mindazok a szempontok, amelyek szerint a felhasználók döntenek az általuk használt operációs rendszer mellett: felület, kezelhetőség, megbízhatóság, gazdaságosság, támogatás, gyártó, stb. Ezen kategóriák közül csak az operációs rendszerek felhasználói felület szerinti csoportosítását ismertetjük, eszerint megkülönböztetünk karakteres és grafikus felületű rendszereket.

Karakteres felületű (*CLI: Command Line Interface*) rendszerek esetében a parancskiadás eszköze a billentyűzet, a számítógép számára az elvégzendő utasításokat egy általában zárt nyelvi struktúra (utasítás-készlet) kifejezéseinek a begépelésével lehet megadni.

A **grafikus felületű** (*GUI: Graphical User Interface*) rendszerek esetében a felhasználói műveletek kiválasztása valamilyen előre definiált grafikus szimbólumkészlet és egy pozícionáló eszköz együttes használatával lehetséges. Az előbbi csoportba tartozó operációs rendszereket szokás még parancsvezérelt, az utóbbiakat pedig eseményvezérelt rendszereknek is nevezni – ez esetben nem a felhasználói felület külalakja, hanem a felhasználói munkavégzést támogató eszközrendszer képezi a csoportosítás alapját. (A parancsvezérelt elnevezés egyértelmű, az eseményvezérelt elnevezés magyarázata pedig az, hogy a grafikus felületű rendszerekben a grafikus környezet változásait (a kijelölő eszköz – pl. az egér – elmozdítását vagy aktiválását (kattintás), vagy egy grafikus szimbólum kiválasztását) eseményeknek nevezik.)

Lényegesen egzaktabb kategorizálásra ad lehetőséget az operációs rendszert futtató számítógép **hardver architektúrája** alapján történő csoportosítás. Ilyen értelemben beszélhetünk a különböző *számítógép-kategóriák* (mainframe kategóriájú számítógépek, szerver számítógépek, személyi számítógépek (munkaállomások), kézi számítógépek, stb.) jellemző operációs rendszereiről, vagy a számítógép *hardver felépítésének* támogatásán alapuló kategorizálásról (processzor-architektúra szerint (CISC vagy RISC alapú rendszerek) egy- vagy többprocesszoros rendszerek, 16-32-64 bites rendszerek, stb.).

Az operációs rendszerek csoportosításának következő eszköze lehet az egyes rendszerek **működésbeli sajátossága** is. Ilyen értelemben vizsgálhatjuk az operációs

rendszerben futó programok számát és együttműködésének módját, a rendszer által kiszolgálható felhasználók számát és az erőforrások megosztásának lehetőségeit, stb.

Az **egy időben futtatható alkalmazások száma** szerint megkülönböztetünk *egyfeladatos (mono-módú vagy kötegelt)* és *többfeladatos (multitasking)* rendszereket – a hangsúly ebben az esetben az egyidejűségen van: a kötegelt rendszerekben egy program futásának ideje alatt újabb program indítására nincs lehetőség, míg a multiprogramozott rendszerekben egyszerre (egymással párhuzamosan) több program is futtatható. (Azonban nem szabad elfeledkezni arról, hogy a Neumann-architektúrájú számítógépeken a műveletvégzés minden esetben szigorúan soros, így a multiprogramozott operációs rendszerek esetében is csak virtuális párhuzamosságról beszélhetünk – azaz a felhasználó számára úgy érzékelhető, mintha programjai egyszerre működnének, valójában azonban csak az egyes programrészek végrehajtása közötti gyors váltásokról van szó.)

A **felhasználók száma szerint** egy- és több-felhasználós rendszereket különböztethetünk meg aszerint, hogy az operációs rendszer rendelkezik-e olyan azonosítási szolgáltatással, amely lehetővé teszi a számítógéppel dolgozó felhasználók (és munkájuk) megkülönböztetését. Az *egy-felhasználós („single user”)* rendszerekben semmilyen azonosítási rendszer nem működik, így nem eldönthető, hogy az adott számítógép egyes erőforrásaihoz ki és milyen jogokkal férhet hozzá. Egy *több-felhasználós („multi-user”)* operációs rendszer esetében minden felhasználói tevékenység kiegészül az elvégző felhasználó azonosítójával és ilyen módon (megfelelő biztonsági rendszer kialakításával) az ugyanazon a számítógépen dolgozó felhasználók tevékenysége teljesen elkülöníthető egymástól. (A több-felhasználós rendszerek esetében célszerű további különbséget tenni aszerint, hogy a különböző felhasználók egyidejűleg kezelhetik-e a számítógépet – amennyiben igen, akkor *hálózati operációs rendszerről (NOS: Networking Operating System)* beszélünk.)

Az operációs rendszerek működésbeli jellemzői alapján történő csoportosításának következő szempontja az **operációs rendszer válaszideje** lehet. A több-folyamatos rendszereknél említettük, hogy csupán virtuális párhuzamosságról van szó, amely mögött alapvetően kétféle technikai megoldás képzelhető el. Az *időosztásos elven* működő *(„time sharing”)* operációs rendszerek esetében a rendszerben futó folyamatok között szétosztásra kerül a rendszer teljes működési ideje: minden folyamat kap egy „időszeletet”, ennyi ideig használhatja a processzort („futhat”), majd át kell adnia a következő folyamat számára, és ez a tevékenység-sorozat mindaddig ismétlődik, míg minden program be nem fejeződik. A *valósidejű (RTOS: Real Time OS)* rendszerek esetében az egyes programok azonnal végrehajtódnak, így a futásuk eredménye („a válasz”) is azonnal (vagy legalábbis emberi léptékkel mérve reális időn – legfeljebb néhány másodperc – belül) rendelkezésre áll. (Természetesen egy valós idejű rendszerben sincs lehetőség két tevékenység egyszerre történő végrehajtására, így amennyiben két program akar ugyanazon időpillanatban elindulni, az egyiknek meg kell várni a másik befejeződését – ebből viszont az is következik, hogy az ilyen rendszerek hatékony működésének feltétele a kis méretű (és ebből következően gyorsan végrehajtódó) programok alkalmazása. A valós idejű rendszereket jellemzően olyan helyzetekben alkalmaznak (többnyire felügyeleti szerepben) ahol az idő szerepe kritikus, mint pl. egészségügy, atomenergia, folyamat-irányítási rendszerek, stb.)

Természetesen ez a felsorolás közel sem teljes és egyáltalán nem véges. Az operációs rendszerek csoportosításával, az egyes rendszerek jellemzésével foglalkozó honlapok közül néhány címe megtalálható a felhasznált források között.

Ellenőrző kérdések

1. Ismertesse az operációs rendszerek fejlődésének fontosabb állomásait!
2. Melyek az operációs rendszerek csoportosítása során alkalmazott legfontosabb szempontok?
3. Csoportosítsa (példákon keresztül) az operációs rendszereket
    a. felület,
    b. architektúra,
    c. működési jellemzők szempontjából!
4. Jellemezze a kötegelt módú operációs rendszerek működését!
5. Ismertesse a többfeladatos operációs rendszerek működésének alapelvét!
6. Hasonlítsa össze a karakteres és a grafikus felületű operációs rendszereket: példák segítségével mutassa be előnyös illetve hátrányos tulajdonságaikat!
7. Hogyan kategorizálhatjuk az operációs rendszereket a felhasználók kiszolgálása alapján?
8. Véleménye szerint melyek azok a területek, amelyek az időosztásos, és melyek azok, amelyeknél a valós idejű válaszidővel dolgozó operációs rendszerek használata a célszerűbb?

2. AZ OPERÁCIÓS RENDSZEREK MŰKÖDÉSE

A számítástechnikai eszközök mind felépítésükben, mind működésük alapvető jellemzőiben számos közös vonást mutatnak – gondoljunk csak a Neumann-elvekre (szerkezetileg: számítógép=CPU+ALU+MEM+I/O, működésileg: tárolt program és soros műveletvégzés) –, így aligha meglepő, hogy az operációs rendszerek többé-kevésbé hasonló elvi szerkezettel rendelkeznek. (Ez ugyan látszólag ellentmond annak, hogy az előzőekben éppen az operációs rendszerek viszonylag nagy számát és az egyes rendszerek között megfigyelhető különbségeket vizsgáltuk – hangsúlyozni szeretnénk azonban, hogy ebben a részben **alapelvekről** van szó. Olyan elméleti megfontolásokról és módszerekről, amelyek az operációs rendszerek helye és szerepe miatt valamilyen formában szinte minden gyakorlati megvalósításban (egy konkrét operációs rendszerben) megtalálhatóak, csak éppen más-más súllyal vesznek részt az egész rendszer működtetésében.)

Az operációs rendszerekre vonatkozóan többféle definíció is létezik:

- olyan program(rendszer), amely felügyeli és vezérli a számítógépen futó valamennyi folyamatot (az ISO szabvány szerinti megfogalmazás)
- a számítógépet alkotó hardver eszközök működését felügyelő és vezérlő program (technológiai megközelítés)
- a számítógép tevékenységét meghatározó programokat felügyelő és vezérlő szoftver (funkcionális megközelítés)
- (a számítógépes rendszerben rendelkezésre álló) erőforrásokat elosztó szuperfolyamat (folyamat-centrikus szemlélet)
- olyan program, amely kapcsolatot teremt (és tart fent) a számítógépet alkotó technikai-technológiai (hardver) elemek és a (számítógéppel tevékenységet végző) felhasználó között (felhasználói szemléletű definíció).

A fentiekkel kapcsolatban két dolgot kell kiemelnünk. Egyrészt az egyes megfogalmazások között (első olvasásra legalábbis) nincs sok különbség (sőt, levonhatjuk azt a tanulságot, hogy az operációs rendszer elsősorban „felügyel és vezérel”, hiszen ez szinte az összes definícióban szerepel), azonban minden megközelítés az operációs rendszer működésének más és más jellemzőjét emeli ki. Másrészt a fenti definíciók alapján az is jól látható, hogy az operációs rendszernek (a felügyeleti feladatokon túl) számos egyéb feladat ellátására is képesnek kell lennie, és hogy ezek a feladatok vonatkozhatnak akár a számítógép hardver elemeivel kapcsolatos műveletekre, akár a felhasználó igényinek kiszolgálására – következésképpen az operációs rendszernek egyfajta közvetítő szerepet is be kell töltenie (a felhasználó és a hardver között).

Az operációs rendszerek legfontosabb alapfeladatai a következők:

- kapcsolattartás a felhasználóval (felhasználói felület),
- jogosultsági rendszer kezelése (pl. hozzáférési jogok, folyamatok prioritása),
- állományszervezés,
- a számítógépet alkotó hardvereszközök különbözőségének „elfedése”, egységesítése (és ilyen módon a felhasználó megkímélése a közvetlen hardverkezeléssel járó többletfeladatoktól – ez az ún. „virtuális gép koncepció”),
- erőforrás-gazdálkodás (memória-kezelés, processzor-ütemezési feladatok, perifériák kezelése),
- programkészítés támogatása (szerkesztő-, fordító-, betöltő modulok),
- állapot-felügyelet (hibakezelés, rendszernaplók),

- (hálózati szolgáltatások: vitatható, hogy önálló feladatnak tekinthető-e, de a mai hálózat-centrikus informatikai modellben egy olyan komplex feladat, mint egy hálózati kommunikációs folyamat kiszolgálása, nem igazából illeszthető be egyik előző csoportba sem), stb.

Összefoglalva tehát azt mondhatjuk, hogy az operációs rendszer egy olyan programrendszer, amely felügyeli a számítógép működését és biztosítja a szükséges kommunikációt a felhasználó és a hardver eszközök között. Programrendszer (nem egyetlen program), hiszen szerkezetileg több önálló (egy vagy több alapfeladat kiszolgálásában közreműködő) komponensből tevődik össze (vö. operációs rendszerek fejlődése, spooling és multitasking rendszerek).

2.1. Az operációs rendszerek szerkezete

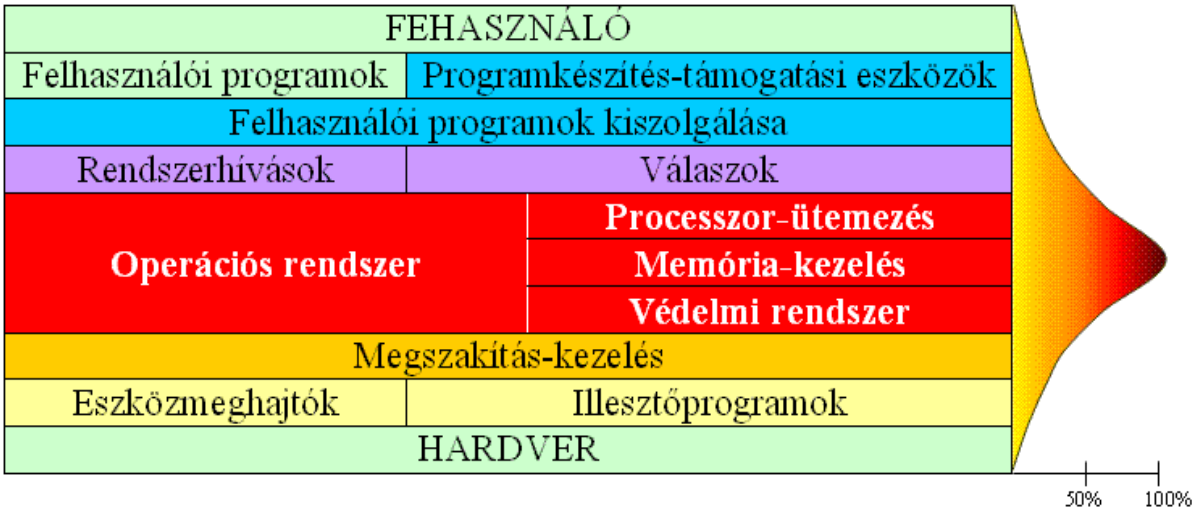
Az operációs rendszert alkotó szolgáltatások közül azokat, amelyek a hardvereszközök kezelését végzik, az operációs rendszer „kernel”-jének (magyarul leginkább „rendszermag”-nak fordítják), a felhasználói felülettel kapcsolatos szolgáltatásokat pedig az operációs rendszer „shell”-jének (magyarul „burok” vagy „héj”) nevezzük.

(A kernel és a shell különválasztásának létezik egy másik értelmezése is: ha nem a szolgáltatás igénylője (hardver vagy felhasználó), hanem a szolgáltatás és a rendszer működése közti összefüggés alapján csoportosítjuk a tevékenységeket. Ez esetben a rendszer működésének biztosítása szempontjából kritikus komponensek alkotják a kernelt, a „kevésbé fontos” (pl. meghibásodás esetén a rendszer működését nem (vagy csak kis mértékben) befolyásoló) komponensek pedig a shellt. Nyilvánvaló, hogy a kétfajta értelmezésben vannak átfedések, hiszen a számítógép alapvető hardver eszközeinek a kezelése (memória, processzor) mindkét esetben a kernel feladata, a felhasználói tevékenységek mindkét esetben a shellre tartoznak – a különbség leginkább a két réteg határán megjelenő szolgáltatások besorolásában van: míg a periféria-kezelő (pl. nyomtatás-vezérlő) komponensek az első esetben kernel, a másodikban shell szinten helyezkednek el.)

2.1.1. Rétegmodell

Mindezeket figyelembe véve beláthatjuk, hogy a számítógépes architektúrák és az operációs rendszerek különbözősége ellenére az egyes rendszerek rendelkeznek egy többé-kevésbé egységes szerkezettel. Ezt a szerkezeti sémát az „operációs rendszer elvi rétegmodelljének” nevezzük.

(A rétegmodell az informatikában gyakran alkalmazott ábrázolási technika egy rendszer vagy folyamat összefüggéseinek leírására, ld. pl. ISO/OSI hálózati rétegmodell, vagy az adatbázis-kezelő rendszerek ANSI/SPARC modellje. Alapgondolata a modularitás: a rétegeket alkotó tevékenységek belső szerkezete nem specifikált, csupán a rétegek közötti kommunikáció szabályai (interfészek). Ilyen módon biztosított a lehetősége annak, hogy az egyes rétegben szereplő komponensek igény szerint helyettesíthetőek (kicserélhetőek) legyenek egy ugyanolyan szerepű másik eszközzel. Az operációs rendszerek esetében a rétegmodellnek a következő ábrán bemutatottól különböző változatai is ismertek, ezek azonban tartalmilag megegyeznek az itt közölt ábrával, csupán az egyes komponensek különböző részletességben továbbbontott vagy összevont változatait tartalmazzák.)



2-1. ábra: a rétegmodell elvi szerkezete

Mit látunk a 2-1. ábrán?

A különböző színek az operációs rendszer különböző rétegeit jelölik: a zölddel jelölt részek nem az operációs rendszer rétegei, a kernel és a shell értelmezések különbsége alapján a további rétegek színjelölése az alábbiak szerint alakul:

- az igénylő szerinti értelmezés szerint a kékkel jelölt részek alkotják a shellt, az összes többi színnel jelölt szolgáltatás pedig a kernelhez tartozik,
- a kritikus működésen alapuló megkülönböztetés szerint a pirossal jelölt részek kernel szintű, a kékkel és sárgával jelölt részek a shell szintű szolgáltatások, a kevert színek (lila és narancs) pedig a rétegeket összekapcsoló szolgáltatásokat jelölik.

A rétegmodell két szélén szerepelnek az operációs rendszer szolgáltatásainak igénylői (a felhasználó illetve az egyes hardver elemek), ezektől a modell belső részei felé haladva előbb kiszolgáló (felhasználói felület illetve az eszközmeghajtók, illesztőprogramok), majd kommunikációs (rendszerhívások illetve megszakítások) és legbelül végrehajtó szolgáltatások találhatók. Vegyük észre, hogy a modell a kernel-szolgáltatásokra szimmetrikus: az operációs rendszer számára teljesen mindegy, hogy az elvégzendő feladatra vonatkozó kérés a felhasználótól vagy egy hardveregységtől származik, a végrehajtás menete többé-kevésbé azonos: a kiszolgáló szolgáltatások érzékelik, a kommunikációs szolgáltatások értelmezik (és szükség szerint rangsorolják – ld. később), a végrehajtó szolgáltatások pedig elvégzik a kijelölt feladatot, majd az eredményre vonatkozó információval ugyanez a tevékenységsorozat fordított irányban is lejátszódik.

Ebből persze az is következik, hogy minél inkább közelítünk az operációs rendszer magját képező szolgáltatások felé, annál nagyobb lesz az operációs rendszer által elvégzendő tevékenységek aránya: míg a két szélen az igénylő tevékenysége a meghatározó, a kommunikációs rétegben ez nagyjából fele-fele, a kernelnél pedig egyértelműen rendszertevékenységekről van szó – ezeket az arányokat olvashatjuk le az ábra jobb szélén látható terület-diagramról.

2.1.2. Megszakítási rendszer

Az operációs rendszer számára mindazok a jelzések, amelyek a számítógépes rendszer működésében, állapotában vagy környezetében bekövetkezett változásokkal kapcsolatos információt hordoznak, rendszerhívásokként jelennek meg. A rendszerhívások tulajdonképpen valamilyen eszköz állapotában bekövetkezett változást jeleznek, és gyakran

magukban hordozzák a kezelésükhöz szükséges operációs rendszer tevékenység azonosításához szükséges információt is. (Pl. ha lenyomunk egy billentyűt, akkor ezzel nem csak azt jelezzük az operációs rendszernek, hogy az egyik periféria állapota megváltozott, hanem azt is, hogy ezzel az eseménnyel egy olyan kezelőrutinnak kell foglalkoznia, amely képes a lenyomott billentyű kódját egy átmeneti tárolóba (puffertár) helyezni.)

A rendszerhívásokat forrásuk szerint három csoportba oszthatjuk: a megszakításokat valamilyen hardver egység generálja, a csapdák a felhasználói programok által létrehozott kérések, míg a kivételek az operációs rendszer szolgáltatásainak a jelzései.

Az előzőekben láthattuk, hogy az operációs rendszer számára a felhasználói és a hardver eszközöktől származó kérések, jelzések kiszolgálásának menete többé-kevésbé azonos (sőt, a rendszerfolyamatok kezelése is ugyanezen elvek szerint valósul meg). Az operációs rendszer megszakítási rendszert kezelő szolgáltatása számára az elvégzendő tevékenység szempontjából nincs különbség a kivétel, a megszakítás és a csapda között: mindhárom esetben ugyanazt a tevékenység-sorozatot kell elvégeznie: igény fogadása – elbírálás – kiszolgálás.

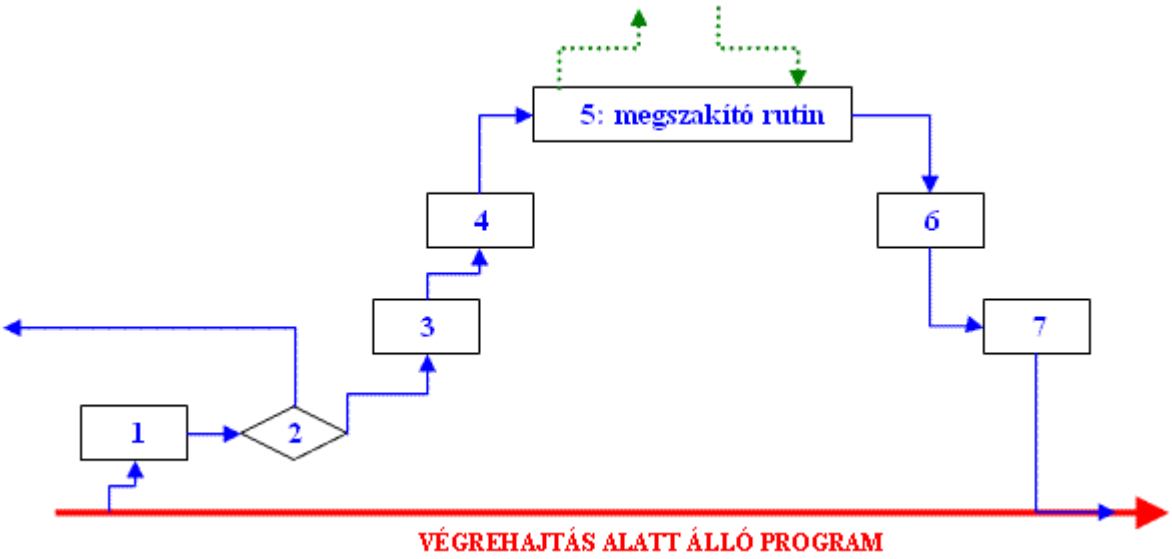
Az „igény fogadása” a megfelelő rétegből érkező jelzés vételét, és ez alapján a kérés kiszolgálásához szükséges operációs rendszer folyamat meghatározását jelenti. (Technikailag az operációs rendszerek gyakran „megszakítási táblázatokat” alkalmaznak: az első oszlopban a megszakítás sorszáma, második oszlopban az adott kérés kiszolgálásáért felelős operációs rendszer rutinjának memóriabeli kezdőcíme szerepel.)

Az „elbírálás” során alapvetően két dolgot kell a megszakítási rendszernek mérlegelnie: a környezetváltás (azaz a felfüggesztett program folytatásához szükséges adatok tárolásának) lehetőségét és a rendszer pillanatnyi állapota valamint a megszakítást kiszolgáló rutin prioritásának viszonyát.

Amennyiben minden feltétel teljesül, akkor jöhet a „kiszolgálás”: a kért rutin futtatása.

Látszólag egyszerű tevékenységről van szó, de vegyünk szemügyre a végrehajtás menetét! Tételezzük fel, hogy a rendszer vizsgálatának pillanatában éppen végrehajtás alatt áll egy program, amikor is érkezik egy megszakítási kérés. Az operációs rendszer fogadja a kérést, majd a megszakítási vektor segítségével meghatározza a kérés kiszolgálásáért felelős rutin címét. A rutin címének ismeretében eldönthető, hogy a kiszolgáló rutin és a jelenleg futó folyamat prioritási szintje hogyan viszonyul egymáshoz: amennyiben a rutin magasabb prioritással rendelkezik, a végrehajtás alatt álló programot az operációs rendszer megszakítja, a rutin elvégzésének idejére, ellenkező esetben a rutin bekerül a végrehajtásra váró folyamatok sorába. Ha a prioritás szerint a kiszolgáló rutin végrehajtása lehetséges, akkor az operációs rendszer (miután befejezte a jelenleg futó program aktuális utasításának végrehajtását – ld. gépi ciklus) elmenti a futó program környezetét: mindazokat az adatokat, amelyek ahhoz szükségesek, hogy a későbbiekben a program a jelenlegi helyéről folytatható legyen. (Ebbe beletartoznak a regiszterek pillanatnyi értékei, a program által használt erőforrások leíró információi (memóriaterületek, használt perifériák, stb.), a program környezetének értékei (pl. a prioritási szintje), stb.. Ezután következik a kiszolgáló rutin környezetének kialakítása, majd a kiszolgáló rutin végrehajtása. A rutin befejezése után az előzőleg elmentett adatok visszaállításra kerülnek, az előzőleg félbeszakított program futása pedig folytatódhat. És így tovább…

A 2-2. ábrán a megszakítási rendszer működésének sematikus ábrája látható, az egyes sorszámok a következő tevékenységeket jelölik:



2-2. ábra: megszakítási rendszer

1. megszakítási kérelem fogadása
2. elbírálás
3. (végrehajtás alatt álló program) futási környezet(ének) mentése
4. (megszakítást kérő rutinnak megfelelő) prioritási szint beállítása
5. megszakítás kiszolgálása (végrehajtás)
6. prioritási szint visszaállítása (az eredetileg futó folyamatnak megfelelően)
7. (3. lépésben elmentett) futási környezet visszaállítása

A fenti tevékenységsorozattal kapcsolatban vegyük észre, hogy a folyamat ismétlődhet („egymásba ágyazódhat”): pl. egy kiszolgáló rutin megszakít egy felhasználói programot, majd a rutin végrehajtása közben érkezik egy újabb (és magasabb prioritással rendelkező) megszakítási kérés, ekkor a kiszolgáló rutin is felfüggesztésre kerül, stb. Az, hogy a folyamat nem végtelen, azt elsődlegesen a prioritási szintek véges számossága garantálja. (Minden folyamatot csak egy nála fontosabb – magasabb prioritással rendelkező – folyamat szakíthat meg, így előbb-utóbb bekövetkezik az az állapot, amikor már olyan prioritású tevékenység végrehajtása zajlik, amit már nem képes más folyamat félbeszakítani. (A prioritási szintek beállítását szokás „maszkolásnak” nevezni.)

(A másik megjegyzés a megszakítási rendszer működésével kapcsolatban a környezetváltáshoz szükséges adatok tárolásának kérdésével kapcsolatos: vegyük észre, hogy minden egyes megszakítás során a folytatáshoz szükséges adatok elmentésre kerülnek – ad absurdum rosszul megválasztott környezeti információk esetében előfordulhat olyan eset, hogy a futó folyamat megszakítása ugyan a prioritási szintek miatt szükséges, de a visszaállítása már nem lesz lehetséges, mivel a környezeti információi elmentésére nincs elegendő tárterület.)

2.2. Kernel funkciók

A kernel, mint láttuk az előző fejezetben, az operációs rendszer alapvető, legbelső része, igaz vele se felhasználó, se alkalmazást készítő programozó nem találkozik. Alapvetően a felhasználói programok és a hardver elemek között helyezkedik el, bár láthatunk arra is példát hogy az operációs rendszer egyes részei a kernel nélkül is elérhetnek eszközöket. Ez azért lehet szükséges, hogy egyes hosszan tartó folyamatok ne terheljék a processzort. A kernel szupervízor módban fut, és az operatív tárban, mint rezidens program foglal helyet az

operációs rendszer betöltése után. Az operációs rendszer más részei, a kiszolgáló programok csak szükség esetén töltődnek be.

2.2.1. Folyamat-vezérlés

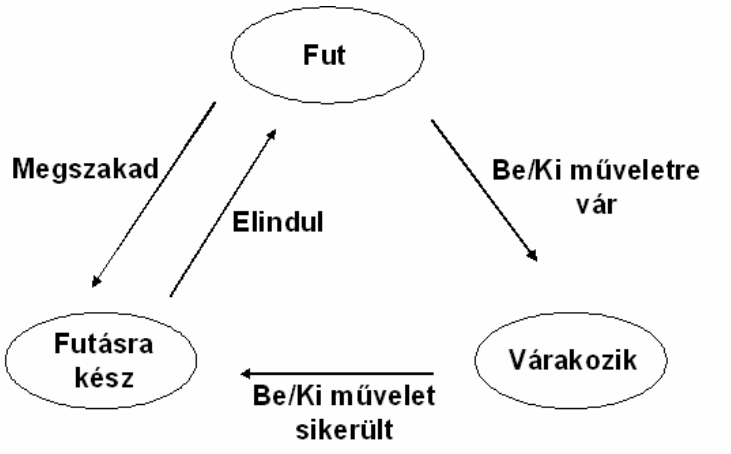
Egy tipikus operációs rendszerben a felhasználói felület kezelése és az alkalmazások kiszolgálása érdekében a kernel az alábbi funkciókat biztosítja:

- megszakítás és kivételkezelés,
- folyamatvezérlés,
- be/kimeneti eszközök vezérlése,
- memóriakezelés,
- a kernelben futó programok helyes működésének felügyelete,
- a szándékos és a véletlen támadásokkal szembeni védelem.

A kernel funkciók együttműködése teszi lehetővé hogy az operációs rendszer betöltse feladatát, a számítógép erőforrásainak menedzselését. Ezek az erőforrások a CPU, a memória, tágabb értelemben a fájlok és a különböző perifériák.

Ahhoz hogy a CPU kihasználtságát növelni tudjuk, és egy időben több programot tudjunk rajta futtatni, a CPU-t meg kell osztani a programok között. Például ha az egyik program B/K műveletre vár, célszerű, hogy egy másik program pedig a CPU-t használja. A harmadik generációs számítógépeknél ez volt az első lépés, hogy kialakulhassanak a multitaszkos operációsrendszerek. Most tekintsük át, ennek elméleti alapjait, hogyan tudjuk ezeket a párhuzamosan futónak tűnő folyamatokat egymástól megvédeni, hogyan tud egy processzor „egy időben” több folyamatot kiszolgálni.

A folyamat elemi lépések sorozata. Egy elemi lépés egy pillanatnyi állapotból eljutni a következő állapotba. A számítógép, mint állapotautomata, determinisztikus, az egyik állapotból a következő állapotba a következőképpen jut. A gépi kódú utasítások sorozatából az aktuális programlépés, a CPU regisztereinek tartalma, az aktuális program változóinak értékei a memóriában, a memória állapota együttesen határozza meg az automatánk következő állapotát. Könnyen belátható, hogy az így leírt folyamat a programhoz, mint gépi kódú utasítássorozathoz, egyértelműen hozzárendelhető. Előfordulhat azonban, hogy egy program viszont több folyamatot indít.



2-3. ábra: A folyamat állapotai és átmenetei

A folyamat állapotai:

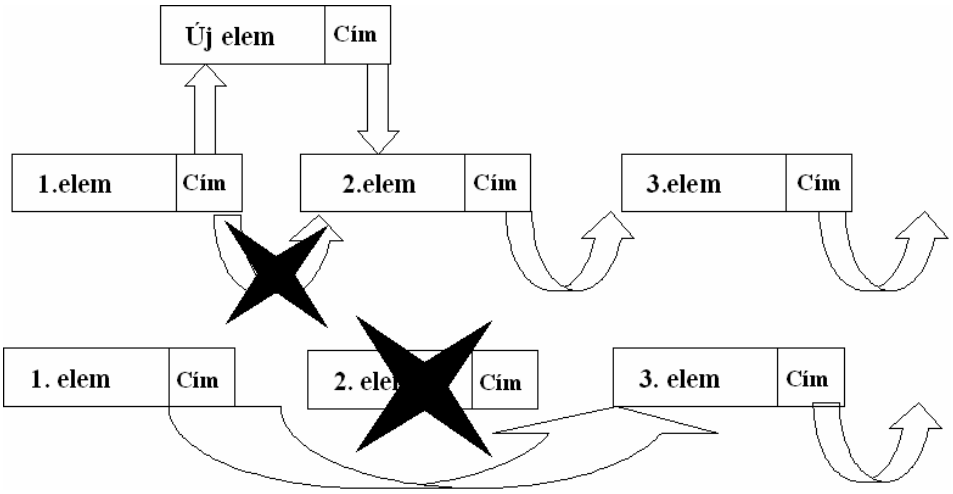
- **Futásra kész**: a memóriába töltés utáni állapot.
- **Fut**: a processzor éppen kiszolgálja.
- **Várakozik**: egy B/K nem áll rendelkezésére.

Hogy a folyamatok működtetését megérthessük még további ismeretek megszerzésére van szükségünk. Ezek a **főütemező** (high-level sheduler), a **folyamatleíró tábla** (PCB azaz process control block), a **várakozási lista** és a **környezetváltás**.

A **főütemező** dönti el, hogy mikor tudja a felhasználó által elindított programot a memóriába tölteni, ha ennek idejét érzi, be is tölti, és egyben a memóriában elhelyezi a folyamatleíró táblát is. Ha a folyamat majd működhet, akkor ez a leírótábla kerül a processzor regisztereibe is.

A **folyamatleíró tábla** tartalmazza mindazt az információt, ami a folyamat működéséhez szükséges[11].

A lista olyan adatszerkezet, amely speciális elemek sorrendi tárolására alkalmas. Egy elem egy memória címből és egy adatelemből áll. A memória cím a memóriában esetlegesen elhelyezkedő listaelemek közül a sorban következő elem címét tárolja, ill. a sor utolsó eleme „Null” értéket. Elég az első elem címét ismerni, hogy a listát sorban olvassuk, valamint a címelem tartalmának cserélgetésével könnyen fűzhetünk hozzá új, vagy törölhetünk régi elemet *(2-4. ábra)*. A **várakozási lista** adatelemei a folyamat leíró táblák.



2-4. ábra: Beszúrás listába, törlés listából

A **környezetváltás** tevékenységén azt értjük, hogy a folyamat leíró tábla adatait a főütemező a memóriából áthelyezi a processzor regiszterébe, így biztosítva a folyamat végrehajtásához szükséges környezetet. Természetesen, ha az éppen a CPU-ban tárolt környezetre a hozzátartozó folyamatnak, mivel még nem ért véget, a továbbiakban is szüksége van, a főütemező környezetváltáskor azt elmenti, azaz visszahelyezi a várakozási listába.

A folyamat átmeneti állapotai:

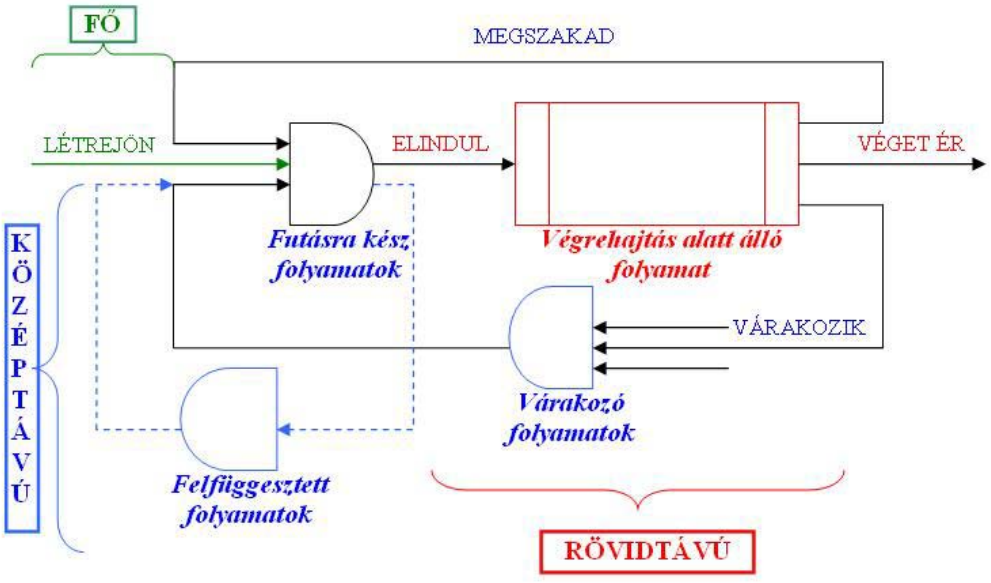
- **Elindul**: a folyamatleíró tábla a memóriából a processzor regiszterébe kerül, a program végrehajtása elkezdődik, ill. folytatódik.

[11] Knapp [1998], 51. o.

- **Megszakad**: csak egyes operációs rendszerek engedik meg, és csak megszakítható (preemptiv) folyamatok esetén fordulhat elő. A folyamat a neki engedélyezett időt kihasználta, visszakerül a futásra kész állapotba.
- **Egy B/K műveletre vár**: ha a futó folyamatnak egy olyan erőforrásra van szüksége, ami az adott pillanatban foglalt, akkor a folyamat várakozó listára kerül, és átadja a helyét egy futásra kész állapotban lévő, vagy egy várakozó folyamatnak.
- **Egy B/K művelet sikerült**: A B/K művelet befejezése után a folyamat visszakerül a futásra kész állapotba.

Visszatérve a fejezet elején található kérdésre, hogyan futtat egy Neumann elvű processzor több folyamatot egy időben, a válasz egyszerű: *sehogy*. Csak gyorsan cserélgeti az aktuálisan futó folyamatokat, a számítógép felhasználójában azt az érzetet keltve, hogy sok folyamat fut párhuzamosan.

A továbbiakban nézzük át milyen problémák, megoldások, stratégiák léteznek az előbb leírt módszerrel történő folyamatcserélgetéssel kapcsolatban. A megszakításokkal az előző fejezet foglalkozik, de előfordulhat két fajta megszakítás, amit itt érdemes megemlíteni. Az egyik az időosztáson alapuló operációs rendszerek esetén fordul elő. Ebben az esetben a főütemező minden egyes folyamatnak azonos időkeretet biztosít (10-20 msec körüli időtartományban), így igyekezve az egyes folyamatok és a rendelkezésre álló erőforrások között egyensúlyt teremteni. A másik, amikor az operációs rendszer rendelkezik egy közbülső ütemezővel (medium level sheduler – *középtávú ütemező*). Ekkor egy külön várakozási lista is keletkezik a felfüggesztett folyamatok leíró táblái részére, a futásra kész folyamatok listáján kívül. E közbülső ütemező nélkül elég egy lista. A stratégiai kérdések elemzése az alacsony szintű ütemező (low level sheduler – *rövidtávú ütemező*) működésének ismertetetésével lehetséges (*2-5. ábra*).



2-5. ábra: Folyamat-állapotok és ütemezők

Mit látunk az ábrán?

A végrehajtásra váró programhoz a főütemező folyamatleíró blokkot rendel, majd az immár folyamat a „Futásra kész folyamatok” várakozási sorba kerül. Innen az alacsony szintű ütemező kiveszi és átadja végrehajtásra a processzornak, azaz a folyamat „elindul”. A

folyamat mindaddig futhat, amíg le nem áll magától („véget ér”), vagy az operációs rendszer le nem állítja („megszakad”). Ha a folyamat azért áll le, mert „várakozik” egy külső forrásból származó adatra, akkor ez idő alatt átadhatja helyét egy másik folyamatnak (ő maga pedig átkerül a „várakozó folyamatok” sorba, a szükséges adatok megérkezésekor pedig visszakerül a „futásra kész folyamatok” közé és kezdődik az egész elölről). A középtávú ütemező folyamatosan ellenőrzi a várakozási sorok állapotát és igyekszik meghatározni, melyik legyen a következő elinduló folyamat ahhoz, hogy a rendszer a legoptimálisabban működjön. Ennek biztosítására arra is képes, hogy egyes folyamatokat kizárjon a processzoridőért történő versengésből („felfüggesztett folyamatok” sor), és csak bizonyos feltételek teljesülése esetén engedélyezze újra a működését.

Annak eldöntésére, hogy az egyes ütemezők milyen szempontok alapján válasszák ki a végrehajtás alá kerülő vagy éppen a végrehajtásból kizárandó folyamatokat, különböző módszerek léteznek, amelyeket röviden **ütemezési stratégiáknak** nevezünk. Az ütemezési stratégiák a folyamat futási idejének bizonyos jellemzőinek elemzésén alapulnak. Egy folyamat (ütemezés szempontjából fontos) tulajdonságai lehetnek pl:

- **Várakozási idő**: a folyamat tétlenségben töltött ideje,
- **Átfutási idő**: a futásra kész állapottól a folyamat befejezéséig terjedő idő,
- **Válaszidő**: a futásra kész állapottól az első futás elkezdéséig eltelt idő.

A gyakorlatban számos megvalósítás létezik, de az a fentiek alapján is érzékelhető, hogy nincs „egyetlen és üdvözítő” megoldás: az egyik hatékonysági jellemző növelésére irányuló eljárás biztosan magában hordozza valamelyik másik jellemző értékének a romlását.

A legegyszerűbb ütemezési algoritmusok a következők:

1. **előbb jött, előbb fut** („First Come First Served”): a folyamatok végrehajtási sorrendje megegyezik a várakozási sorba érkezésük sorrendjével. Előnye, hogy a lista tárolási szerkezetének megfelel a feldolgozás menete, így egyszerűen megvalósítható. Hátránya, hogy a nagyobb időigényű folyamatok feltarthatják a rövidebb folyamatokat (torlódás).
2. **legrövidebb fut** („Shortest Job First”): a futásra várakozó folyamatok közül az kerül sorra, amelynek a legrövidebb a processzoridő-igénye. Előnye, hogy a leggyorsabb folyamat-végrehajtást teszi lehetővé. Hátránya, hogy előfordulhatnak kizárt folyamatok (egy nagy időigényű folyamat örökösen a várakozási sor végén marad, miközben folyton érkeznek újabb, rövidebb folyamatok), valamint hogy a processzoridő-igény előre ismerni kell (az esetek többségében ez nem meghatározható, csak becsülhető érték!).
3. **körben járó** („Round Robin”): minden folyamat rendelkezésére áll egy időszelet, ameddig működhet, ennek leteltével a várakozási sor végére kerül. Előnye, hogy minden folyamat azonos ideig működhet (nem fordulhat elő feltartás vagy kizárás). Hátránya, hogy a folyamatok újra-sorbaállítása miatt arányosan növekszik a környezetváltások (és adminisztrációs műveletek) száma, így nő a folyamatok végrehajtásának ideje is.

A fenti algoritmusokkal együtt (vagy azok helyett) alkalmazhatóak a folyamatok prioritását alapul vevő ütemező stratégiák is.

2.2.2. Memória-kezelés

Az előzőekben megismerkedtünk a folyamatok működésének mechanizmusával, többfolyamatos rendszerekben alkalmazott időzítési módszerekkel. Azonban a folyamatok

nem kizárólagosan a processzornak kiadott utasításokat jelentik, így a folyamatok működésének pontosabb megismeréséhez elengedhetetlen a memória-kezelés vizsgálata is.

A *folyamat-orientált szemlélet* lényege az, hogy az éppen végrehajtás alatt álló tevékenység számára a számítógép úgy jelenik meg, mintha azt kizárólagosan használná, minden erőforrás teljes egészében a rendelkezésére állna. Mind az adatok, mind az utasítások az operatív tárban helyezkednek el, azonban ha több folyamat végrehajtása (látszólag párhuzamosan) zajlik, és minden folyamat saját adatokkal dolgozik, akkor több kérdés is felmerül:

1. **Kernel-folyamatok védelme**: mivel a felhasználói folyamatok vezérlésére szolgálnak, a kernel-folyamatok állandóan a memóriában tartózkodnak, biztosítani kell, hogy más folyamatok ne módosíthassák az ezen folyamatok által használt adatokat.
2. **Folyamatok védelme egymástól**: hasonlóan belátható, hogy az sem szerencsés, ha az egyes folyamatok más folyamatok adatainak vagy utasításainak tárolására szolgáló memóriarészeket képesek módosítani.
3. Egyidejűleg **több folyamat az operatív memóriában**: a folyamatok gyorsabb végrehajtása érdekében (ha lehetséges) célszerű több folyamatot is az operatív memória (elkülönülő) részeiben tartani – a folyamatváltás ilyenkor csak a következő folyamat által használt memóriatartomány kiválasztását jelenti, nem kell várni a virtuális memóriába történő kiírás-visszaolvasás műveletre.
4. **Rendelkezésre álló memóriánál nagyobb memóriaigényű folyamatok kezelése**: akár az előbbi esetben is előfordulhat, de önállóan is jelentkezhet az a probléma, hogy a folyamat utasítás- és adatrésze nem fér el a fizikai memóriában – ilyenkor csak egy része kerül az operatív tárba (annyi, amennyi elfér, és az, amelyik éppen a végrehajtás alatt áll), a többi szükség (hivatkozás, amikor rákerül a végrehajtás menete) esetén töltődik be (az immár feleslegessé vált részek helyett).
5. **Végrehajtás közben kialakuló memóriahivatkozások (címek) kezelése**: az előző folyamatos cserélgetésnek viszont az (is lehet) a következménye, hogy előre nem meghatározható, hogy hol vannak a felhasználandó adatok vagy a következő utasítás – milyen módon határozhatjuk meg?

A megoldást (természetesen ez esetben is) az operációs rendszer (kernel) szolgáltatásai, konkrétan a memória-kezeléssel kapcsolatos tevékenységek jelentik.

Valóságos tárkezelés: a számítógépben rendelkezésre álló fizikai memórián (*operatív tár*) belüli műveletek.

Virtuális memória: a háttértárolón elhelyezkedő, az operatív tárral azonos szerepet betöltő tárolóterület.

Mivel a számítógép csak azokkal az adatokkal és utasításokkal képes műveletet végezni, amelyek az operatív tárban találhatók, ezért a virtuális tárkezelés lényege abban áll, hogy csak az éppen feldolgozás alatt álló folyamat adatai és utasításai helyezkednek el az operatív tárban, és a processzor folyamat-váltásaihoz hasonlóan ezek az információk is folyamatosan cserélődnek az operatív és a virtuális memória között.

Az operatív tár kezelésére szolgáló módszerek[12]:

A **rögzített címzés** lényege, hogy az operációs rendszer a memória rögzített (általában legkisebb sorszámú) részén helyezkedik el, a fennmaradó memóriaterület összefüggően áll a folyamatok rendelkezésére. Az operációs rendszer számára fenntartott terület végét egy **határregiszter** rögzíti, az ellenőrzés mindössze azt jelenti, hogy felhasználói program nem hivatkozhatott ennél a címnél kisebb címre. A programok **rögzített címeket** használhatnak(az utasítás vagy az adat a program minden futása során ugyanabban a memóriarekeszben helyezkedett el).

Az **áthelyezhető címzés** alkalmazása során az operációs rendszer mérete (és így az általa elfoglalt memóriaterület nagysága) nem állandó, így a folyamatok számára rendelkezésre álló memóriaterület kezdőcíme minden futásnál megváltozhatott. Következésképpen a rögzített címzés ennél a módszernél nem alkalmazható, helyette a programok **relatív címek** alkalmazásával hivatkoznak a tárolt információra: a program kódja nem a keresett adat memóriában elfoglalt helyét (rekeszének sorszámát, memóriacímét) tartalmazza, csupán azt, hogy az adott utasítás vagy adat a program *kezdetéhez képest* hol helyezkedik el.

A program elindulásakor az operációs rendszer meghatározza, hogy a folyamat melyik memóriacímtől kezdődően töltődjön be, és ezt az értéket letárolja egy speciálisan erre a célra kialakított tárolóba, amit **bázisregiszternek** neveznek. A folyamat végrehajtása során a *bázisregiszter értéke + hivatkozási (relatív!) cím* képlet alapján kiszámolható a keresett információt tartalmazó memóriarekesz fizikai címe.

Az **átlapoló** („overlay”) **technika** lényege, hogy a programokat logikai részekre (**blokk**) osztjuk, és egy időben nem a teljes program, csak a végrehajtás alatt álló kódrészt tartalmazó blokk tartózkodik az operatív memóriában, a többi a virtuálisban. Előnye, hogy lehetőséget biztosít az operatív memória méreténél nagyobb programok futtatására is.

Az éppen betöltendő blokk kiválasztásához az operációs rendszer nem nyújt támogatást, ezért az aktuálisan használni kívánt modul kiválasztása a programozó feladata. Ennek érdekében a programoknak van egy vezérlő modulja, ami állandóan a memóriában tartózkodik, és eldönti, hogy mely blokkot kell betölteni.

A **tárcsere** („swapping”) módszer szerint az egyes folyamatok közti váltáskor a teljes memória kiírásra kerül a háttértárra („*swap-file”*). Előnye az egyszerűsége, hátránya a nagy mennyiségű adat kiírására-visszaolvasására fordítandó (a futási időhöz képest) hosszú idő. Optimalizálható, ha csak a legutolsó mentés óta megváltozott memóriaterületeket írjuk ki a cserefájlba, ennek az adminisztrálása viszont lényegesen bonyolultabb.

Az **állandó partíciók** kezelésekor az operatív memóriát egymástól független részekre osztják (**partíció**), amelyek mindegyike úgy viselkedik az egyes folyamatok számára, mintha teljes memória lenne. Előnye, hogy egy időben több folyamat is tartózkodhat a memóriában, így a folyamatok közti váltás nem igényel háttértár-műveletet, azaz lényegesen gyorsabb. Hátránya, hogy a partíciók mérete nehezen meghatározható: ha túl kicsi, nem fér el benne a folyamat; ha túl nagy, sok kihasználatlan hely marad az egyes partíciókon belül (amik összességében akár képesek lennének egy újabb folyamat működtetésére) – ezt nevezzük **belső elaprózódásnak**.

[12] Az alább bemutatásra kerülő (csak az operatív memóriát kezelő) módszerek csupán az operációs rendszer működési modelljének megértését hivatottak szemléltetni, az aktuális gyakorlati megvalósítások szinte kivétel nélkül a virtuális memória-kezelési modellt valósítják meg!

A partíciók kezelése során az operációs rendszerre újabb feladatok hárulnak: partíciónként meg kell határozni (és nyilván kell tartani) a partíciók kezdőcímét (bázisregiszter!) és méretét (az optimális kihasználásra törekedve általában különböző méretű partíciókat használnak) és gondoskodni memória hatékony elosztásáról a folyamatok között.

A **rugalmas partíciók** módszere az előző módszer javítása: a partíciók mérete a folyamatok memória-igényének megfelelően kerül meghatározásra (elvben minden folyamat annyi memóriát foglal le, amennyire szüksége van – a gyakorlatban az egyszerűbb adminisztrálás végett általában nem pontosan ekkora, hanem a legközelebbi 2 hatvány méretű partíciók jönnek létre). Az operációs rendszer nyilvántartja az egyes partíciók foglaltságát, a távozó folyamatok után fennmaradt „lyukakból" választ helyet, ahová az érkező folyamat belefér.

Nyilvánvaló, hogy ez a módszer gyakorlatilag kiküszöböli a belső elaprózódást (a partíciók mérete csak alig különbözik a folyamatokétól), viszont előfordulhat, hogy sok kicsi „memórialyuk" keletkezik (*ezek a „lyukak" már nem részei semmilyen partíciónak!, gyakorlatilag felhasználható memóriaterületek lennének, csak sajnos nem összefüggőek*), ez a **külső elaprózódás**. Megoldást jelent a széttagozódást csökkentő algoritmusok futtatása, melyek a memóriaterületek átcsoportosításával egymás mellé rendezik a szabad helyeket, ez azonban sebesség-csökkenéssel jár.

Lapozás (paging): lényege, hogy a folyamatok által használható memóriaterületnek nem kell összefüggőnek lennie. A memóriát viszonylag kis méretű, azonos nagyságú részekre (**lapok**) osztva az egyes folyamatok a szabad lapok közül annyit foglalnak le, amennyire szükségük van. Így mind a belső, mind a külső elaprózódás megszüntethető: a belső elég kis – a gyakorlatban 2-4 Kbájt méretű – lapok használatával, a külső a szabad lapoknak a folyamatok közötti igény szerint elosztásával (nem lesznek kihasználhatatlan „lyukak").

Ezzel a megoldással azonban a címzési eljárás tovább bonyolódik: nem elég azt tudni, hogy a folyamat kezdeti címéhez (bázisregiszter) képest mennyivel eltolva található a keresett információ, de azt is tudni kell, hogy ez a cím hanyadik logikai lapra mutat, illetve annak a lapnak a fizikai memória mely címtartománya felel meg. A probléma megoldására a **laptáblák** szolgálnak.

Laptábla: a folyamat betöltésekor az operációs rendszer által létrehozott leíró tábla, amely a folyamat logikai lapjainak megfelelő fizikai memóriacímeket tartalmazza.

Ezek után a keresett adat címének meghatározása a következőképpen történik: a keresett adat logikai címét 2 részre osztjuk: az első rész megadja a **lapszámot**, a második rész a **lapon belüli eltolás** mértékét. A lapszám alapján a laptáblából meghatározható a lap **fizikai kezdőcíme**, amihez a hagyományos módon hozzáadva a lapon belüli eltolást, meghatározható a keresett memóriarekesz sorszáma.

A **virtuális memóriakezelés** alapelve, hogy a folyamatok számára az operatív memória és a virtuális memória azonos módon használható. A folyamatok ugyan továbbra is csak a valós memóriában működhetnek, de a memóriakezelés szempontjából közömbös, hogy az egyes folyamatok éppen az operatív vagy a virtuális memóriában helyezkednek el. A megvalósítás a „lapozásos" memóriakezelési technikán alapszik, azonban a laptáblában az egyes lapok leírását ki kell egészíteni egy jelzőbittel, amely azt mondja meg, hogy az adott lap az operatív vagy a virtuális memóriában található. Amennyiben olyan lapra érkezik hivatkozás, amely a virtuális memóriában található (ezt nevezik **laphibának** – ez nem jelent valódi hibát!), akkor az operatív memória valamely lapja kikerül a virtuális memóriába, a virtuális memória hivatkozott lapja pedig betöltődik az operatív tárba. A kérdés csupán az,

hogy melyik legyen az a lap, amely kikerül az operatív tárból. Ennek meghatározására szolgálnak a **lapcsere stratégiák**:

1. **Optimális stratégia**: azt a lapot kell kicserélni, amelyre a legtávolabbi jövőben történik hivatkozás – mivel ez nem eldönthető egy folyamat futása során, ez csak elvi stratégia!
2. **Előbb jött, előbb megy** („First In First Out”): azt a lapot kell kicserélni, amelyik a legrégebben tartózkodik az operatív tárban.
3. **Legrégebben használt** („Last Recently Used”): azt a lapot kell kicserélni, amelyre a legrégebben történt hivatkozás.
4. **Második esély** („Second Chance”): módosított FIFO, csak akkor cseréli ki a legrégebben bent levő lapot, ha a folyamat a legutolsó lapcsere óta nem használta. Ehhez az egyes lapokhoz (a laptáblában) hozzárendelünk egy újabb jelzőbitet, amelyet a lap használatakor (ha hivatkozás történik a lapra) 1-esre állítunk, és ha laphiba esetén ez a lap lenne a kicserélendő, akkor a jelzőbitet 0-ra állítva betesszük a FIFO végére (mintha most töltődött volna be), és megnézzük a következő lapot (amíg olyan lapot nem találunk, amelynek a „használt”-bitje 0).
5. **Mostanában használt** („Recently Used”): módosított LRU, csak akkor cseréli ki a legrégebben használtat, ha azt a legutolsó lapcsere óta nem használták. Elve az előző módszerhez hasonlóan a lap használatának jelzésén alapszik, laphiba esetén olyan lapot választunk, amelynek „használt”-bitje 0, de lapcserekor minden operatív tárban levő lap jelzőbitjét kinullázzuk (Így előzhető meg, hogy valamely lap „örökké” a memóriában maradjon.)

A lapcsere stratégiák hatékonyságának összehasonlítására tapasztalati illetve statisztikai úton előállított lapigény-sorozatokat (**referencia-stringek**) használnak, ugyanazon a mintasoron vizsgálva az egyes módszerek által okozott laphibák számát.

2.3. Shell szolgáltatások

Az operációs rendszerek eddig megismert (kernel szintű) szolgáltatásai a felhasználó számára általában kevésbé „érzékelhetőek” (nem látványosan, a felhasználó által kezelhető módon befolyásolják a rendszer működését), mint azok, amelyek a közvetlen beavatkozás lehetőségét (és ilyen módon a rendszer vezérlésének érzetét) adják. A shell-szintű szolgáltatások közül a felhasználói utasításainak feldolgozásával és a felhasználói információk tárolásával kapcsolatos szolgáltatások a legfontosabbak, az előbbit az operációs rendszer parancsértelmező komponense, az utóbbit pedig az állomány-kezelő rendszer végzi. A következőkben ezek alapelveivel ismerkedünk meg.

2.3.1. Felhasználói felületek

2.3.1.1. Parancsmódú rendszerek

Ahogy azt az operációs rendszerek csoportosításakor már láttuk, a parancsvezérelt operációs rendszerek esetében a felhasználónak az egyes tevékenységek kiválasztásához egy utasítás-listából kell kiválasztania az általa elvégezni kívánt tevékenységhez rendelt kulcsszót. Az operációs rendszer által ismert (értelmezhető) parancsok együttese az utasításkészlet. (Egyes operációs rendszerekben ez a „lista” az operációs rendszer elválaszthatatlan (és felhasználói eszközökkel nem bővíthető, nem módosítható) részét képezi: a parancsok megvalósításáért felelős program(ok) állandóan telepítésre kerülő rendszerfájl(ok) formájában vannak jelen a rendszerben – ezeket a parancsokat szokás „belső” parancsoknak is nevezni. A „külső” parancsok (amennyiben az operációs rendszer támogatja) elnevezés azt fejezi ki, hogy

olyan programokról van szó, amelyek részei ugyan az utasításkészletnek, de nem feltétlen képezik a rendszer szerves részét: igény szerint rendelkezésre álló műveleteket megvalósító parancsok, kiegészítések, rendszerszintű segédprogramok tartozhatnak közéjük.)

Utasítás-szerkezet

Az utasításkészlet a parancsok felsorolásán túl azok használatával, megadásának módjával (esetleg feltételével) kapcsolatban is tartalmaz(hat) előírásokat. Ezek közül a legáltalánosabbak a parancsok paraméterezésével foglalkozó megkötések. Az általánosság megszorítása nélkül feltételezhetjük, hogy egy utasítás a következő három szerkezeti elemből épülhet fel:

kulcsszó [paraméterek] [kapcsolók].

A kulcsszó az utasításkészlet eleme: olyan karaktersorozat, amely azonosítja az elvégzendő tevékenységet. Az egyes feladatokat azonban (természetesen) nem minden esetben kell ugyanazokkal az adatokkal és ugyanolyan módon elvégezni. A paraméterek (a parancs aktuális végrehajtása során) a műveletben érintett adatok körét jelölik ki, a kapcsolók pedig a tevékenység végrehajtásának a körülményeit befolyásolják. A fenti példában (az egyébként általánosan használt) terminológiának megfelelően a nem kötelező (opcionális) paramétereket [] (szögletes zárójelek) jelzik. Ennek megfelelően a fenti alak azt jelenti, hogy egy utasítás kiadásához mindenképpen meg kell adni az elvégezni kívánt tevékenységet meghatározó parancsot, és ezen kívül (nem kötelező jelleggel) lehetnek a parancsnak paraméterei, és használhatunk a parancs működését befolyásoló további kapcsolókat is.

Ahhoz, hogy ezt megértsük, nézzük meg a következő (képzeletbeli!) parancsszerkezetet:

PRINT levél –NOHEADER

(A példa szándékosan használ angol kifejezéseket: a parancsvezérelt rendszerek általában nem rendelkeznek nemzeti (nyelvi) támogatással…)

A fenti utasításban a PRINT a nyomtatási művelet parancsa, a levél a nyomtatandó adatállomány azonosítója (paraméter: egy másik állomány nyomtatásakor ezen a helyen egy másik azonosító szerepel), a –NOHEADER pedig azt jelzi, hogy a nyomtatási művelet során a fejlécben szereplő információk megjelenítését nem kérjük (kapcsoló: ha a következő nyomtatási műveletben kérünk fejlécet, akkor nem szerepeltetjük).

Egy másik (immár nem képzeletbeli) példa a Windows operációs rendszer TREE parancsát mutatja be. Az operációs rendszer súgója szerint a parancs teljes alakja a következő:

TREE [meghajtó:][elérési_út] [/F] [/A].

A TREE maga az utasítás (megadása kötelező, szerepe, hogy a megadott – vagy aktuális – elérési útvonalban szereplő könyvtár tartalmát grafikusan megjeleníti), ezt követheti (opcionálisan) a vizsgálni kívánt meghajtó betűjele és a könyvtár elérési útvonala. (Ha ezeket nem adjuk meg, a parancs – alapértelmezés szerint – az aktuális könyvtárra fog vonatkozni.). A /F és a /A kapcsolók (a jelölésből látszik, hogy használatuk szintén nem kötelező), az előbbi hatására a megjelenítésben szerepelni fognak a kiválasztott mappában lévő fájlokat is, az utóbbi pedig szabványos ASCII karaktereket használ (bővített karakterek használata helyett, ilyen módon az eredmény pl. karakteres megjelenítő eszközön is látható).

A fenti parancsszerkezet látszólagos egyszerűsége ellenére is számos kérdést vet fel:

- melyek a kötelező és melyek az opcionális elemek?
- milyen sorrendben kell az egyes elemeket megadni?

- hogyan lehet eldönteni, hogy az utasítás módosítója paraméter vagy kapcsoló?

Általános érvényű válasz ugyan nincs ezekre a kérdésekre, de vannak olyan irányelvek, amelyek az operációs rendszerek többségében (kvázi-szabványként) megtalálhatók:

- az utasításszerkezetnek minden esetben kulcsszóval kell kezdődnie.
- az utasítás végét az operációs rendszer számára egy kitüntetett szimbólum (általában billentyű(kombináció): pl. Enter) jelzi.
- a paraméterek/kapcsolók száma utasításonként eltérő lehet (megengedve a paraméterrel/kapcsolóval nem rendelkező utasítást is).
- amennyiben a paraméterek és kapcsolók sorrendje nem kötött, akkor az adott elem előtt álló szimbóluma dönti el a szerepét (pl. a kapcsolók egy – jellel kezdődnek).
- a több paraméterrel/kapcsolóval rendelkező utasítások esetében az egyes elemeket legalább egy elhatároló szimbólum (általában szóköz) választja el egymástól.

A parancsok kiadása tehát az utasításkészlet és az utasítás-szerkezet ismeretében (viszonylag) egyszerű feladat – ez azonban nem jelenti azt, hogy az eredmény garantáltan hibátlan is. Az operációs rendszer az utasítás lezárása („utasítás-kiadás") után szintaktikailag ellenőrzi a kapott parancsot (ld. fenti szempontrendszer), majd megpróbálja a megadott paraméterek és kapcsolók értékei alapján végrehajtani. A végrehajtás eredményével kapcsolatos információk kezelése szempontjából alapvetően két megközelítés fordul elő a különböző rendszerekben, a pozitív és a negatív nyugtázás. Pozitív nyugtázás alatt azt értjük, hogy az operációs rendszer a sikeresen elvégzett parancs végrehajtásáról ad valamiféle visszajelzést a felhasználónak (általában az elvégzés sikerességének igazolása mellett – amennyiben értelmezhető – az eredménnyel kapcsolatos információt is megjeleníti). A negatív nyugtázásos rendszerben ezzel szemben csak a hibás parancsok váltanak ki reakciót az operációs rendszer részéről – a sikeresen végrehajtott parancsok sikerességét a hibaüzenet hiánya(!), illetve a készenléti jel megjelenése jelzi.

A parancskiadás módszerei

Habár a parancsmódú rendszerek a működés szempontjából jellemzően kötegelt módúak, ez azonban egyrészt nem szükségszerű, másrészt ilyen esetben is biztosítani kell az egymással összefüggő utasítások együttműködését támogató (lehetővé tevő) eszközrendszert. Ez azt jelenti, hogy ugyan az ilyen rendszerekre az interaktív működés jellemző (a felhasználó kiadja a parancsát, az operációs rendszer pedig végrehajtja azt – amíg az utasítás feldolgozása folyik, addig a felhasználónak nincs lehetősége beleavatkozni a rendszer működésébe, nem adhat újabb utasítást), azonban ezek a rendszerek is tartalmaznak olyan eszközöket, amelyek több parancs (többé-kevésbé) egyidejű kiadását is lehetővé teszik.

A legegyszerűbb ilyen eszköz az utasítás-köteg (parancsfájl, script). Az utasítás-köteg egy olyan állomány, amelynek minden sorában egy-egy (egyébként önállóan kiadható és végrehajtható) parancs szerepel. Ezt az utasítás-köteget átadva a parancsértelmezőnek (elindítva) a benne foglalt utasítások mindegyike végrehajtódik, és csak az összes utasítás végrehajtása után kerül vissza a vezérlés a felhasználóhoz. (A felhasználó számára úgy tűnik, mintha az összes utasítás egy lépésben hajtódna végre.)

Kicsit bonyolultabb a helyzet, ha a végrehajtandó programok futtatására azonos időben kell sort keríteni (ld. párhuzamos folyamatok). A parancsmódú rendszerek esetén a programok párhuzamos működésének eszköze az ún. háttérben történő futtatás (egyes rendszerek terminológiájában az ilyen elven működő programokat memória-rezidensnek vagy TSR-módúnak (Terminate and Stay Resident) nevezik). A háttérben történő futtatás alapelve az, hogy a program az elindítása után azonnal (nem csak a tevékenységének befejeztével, mint

az az egyéb programokra jellemző) visszaadja a vezérlést (gyakorlatilag a készenléti jelet) a felhasználónak, így biztosítva számára újabb parancs kiadásának a lehetőségét. (Ebből természetesen az is következik, hogy a háttérben futó program futás közben nem képes kommunikálni a felhasználóval (gyakorlatilag szinte „láthatatlan” a felhasználó számára – erre utal az elnevezése is.)

Parancsok együttes végrehajtására szolgál a csatolás (más rendszerekben csatorna, „pipe”) is – ebben az esetben azonban nem azonos időben történik az egyes parancsok feldolgozása, hanem egymás után (ez még eddig semmiben nem tér el a felhasználói parancskiadás menetétől), azonban a később elindított parancs ebben az esetben hozzáfér az előtte befejeződött parancs eredményéhez, és azt feldolgozhatja. (A működési elvet leginkább egy futószalaghoz lehetne hasonlítani.)

A csatoláshoz (első látásra legalábbis) némiképp hasonló feladatot lát el az operációs rendszerek átirányítási szolgáltatása. Az átirányítás megértéséhez azt kell tudnunk, hogy a karakteres felületű rendszerek alapértelmezés szerint rendelkeznek egy beviteli eszközzel (általában a billentyűzet) és egy megjelenítési eszközzel (általában a monitor). Az egyes utasítások feldolgozása során az operációs rendszer alapfeltevésekkel él: amennyiben az adott programnak külső információra van szüksége, azt a billentyűzetről fogja kapni, illetve ha meg kíván jeleníteni valamit, azt a képernyőre küldi. Az átirányítás műveletével ezeket az alapértelmezéseket bírálhatjuk felül, megadva, hogy az adott folyamat végrehajtása során honnan származzanak a bemeneti adatok és hová kerüljenek az eredmények. Az elv gyakorlati alkalmazására példa a nyomtatás: az eredmény a képernyő helyett a nyomtatóra kerül, de az átirányítás nem csak eszközre, hanem folyamatra (programra) is vonatkozhat (egy program eredménye a képernyő (mint megjelenítő eszköz) helyett egy újabb programhoz kerül (ami pl. további műveletet végez vele).

Az előbbiek magyarázatául (és a karakteres felületű operációs rendszerek parancskezelésével kapcsolatos ismeretek lezárásaként) álljon itt egy példa (a példában a DOS operációs rendszer utasításait és az ebben az operációs rendszerben alkalmazott átirányítási és csatolási szimbólumokat alkalmazzuk):

DIR c:\windows /S /B /A-D | FIND „net” | SORT /R > lista.txt

Ebben az utasításban a „DIR”, a „FIND” és a „SORT” különböző utasítások, amelyeket (közvetlenül az utasítás után, ld. DIR és FIND) paraméterek és a „/” szimbólummal jelölve kapcsolók egészítenek ki, a „|” a csatolás, a „>” pedig az átirányítás jele. A parancs kiadása után az operációs rendszer előbb végrehajtja az első parancsot (elkészít egy listát a Windows könyvtárban található állományokról), majd az eredményt (ahelyett, hogy megjelenítené a képernyőn) átadja feldolgozásra a következő parancsnak (ami megkeresi azokat a sorokat, amelyekben szerepel a megadott szövegrész), végül pedig ennek az eredményét szintén továbbadja egy következő parancsnak (ami történetesen ábécé szerint csökkenő sorrendbe rendezi az előző listát) – majd az eredményt az alapértelmezett kimeneti eszköz (képernyő) helyett (a teljes műveletsor végén) elhelyezi a lista.txt néven létrehozott állományba.

2.3.1.2. Ablakozó (esemény-vezérelt) rendszerek

A karakteres felület hatékony, de körülményes használatát előbb kiegészítendő, majd kiváltandó jelentek meg az első grafikus felületű rendszerek (GUI: Graphical User Interface), amelyek kényelmes (egyszerű) használhatóságuk miatt hamarosan az elterjedt informatikai eszközök elsődleges felhasználói felületévé léptek elő. A grafikus felületű rendszerek működésének hátterében ugyanazok a tevékenységek zajlanak, mint a karakteres felület esetén, csak éppen az eszközrendszer különbözik.

A grafikus rendszerek esetében az elvégezni kívánt tevékenység megadása nem parancsok segítségével történik, hanem a műveletet reprezentáló grafikus szimbólum kiválasztásával. Ebből következik, hogy a grafikus felületű operációs rendszerek használatához szükség van grafikus szimbólumokra (amelyek az egyes tevékenységeket reprezentálják), valamint egy kijelölő eszközre (amellyel az egyes szimbólumok közti választás elvégezhető). Ez utóbbi lehet akár a billentyűzet is, de a grafikus pozícionáló eszközök lényegesen elterjedtebbek: a különböző típusú egerek, érintő-képernyős megoldások, stb. Az ilyen rendszerek működése (mind a felhasználói szinten megjelenő szolgáltatásai, mind az operációs rendszernek a kezelésével kapcsolatos műveletei) és felépítése (megjelenése, a használható eszközök – komponensek – rendszere) többé-kevésbé szabványosnak mondható.

A GUI működési alapelvei

A grafikus felületű operációs rendszerek működési alapelvei lényegében az „X Window System” (röviden X11 vagy X) szabvány valamilyen szintű megvalósításaként állnak elő. Az X11 egy ügyfél-kiszolgáló (kliens-szerver) alapú rendszer: a grafikus képernyőt egy kiszolgáló (szerverprogram) kezeli, és minden program, amely írni vagy rajzolni szeretne a képernyőre – ezek a kliensek (ügyfelek) –, ezzel a szerverrel kommunikál. (A hagyományos ügyfél-kiszolgáló modellel szemben az X11 „fordított” értelemben használja a fogalmakat: a szerver mindig a saját gépen futó grafikus felügyelő komponens, a kliens a programot futtató számítógép, ami lehet lokális, de lehet egy hálózaton keresztül elért másik gép is.)

A szabvány nem tartalmaz sem az ablakok kezelésére, sem azok komponenseire, viselkedésére vonatkozó leírásokat, ezek ellátása egy speciális kliens, az ablakkezelő (*window-manager*) feladata. Alapeleme természetesen az ablak (ablak a képernyőnek az a része, ahol az adott program „fut”, adatokat dolgoz fel, kommunikál a felhasználóval. Egy kliens létrehozhat ablakot a képernyőn, azon belül létrehozhat újabb ablakokat (az „önmagán” belüli területeket megváltoztathatja, de az önmagán kívülieket nem), tehát az ablakok egy hierarchikus struktúrát alkotnak (a struktúra legmagasabb szintű eleme a „root” ablak, ezen belül találhatók a „közönséges” ablakok, amelynek további (funkcionális) részeket tartalmazhatnak: üzenet-ablakot, menüt, ikonokat, gördítő-ablakot). Az ablakok közül mindig pontosan 1 lehet „aktív” – a műveletvégzés szempontjából kiválasztott.

A szabvány definiálja még a kommunikációs eszközök kezelésével kapcsolatos módszereket is: az egér viselkedését (csak érdekességképp: 5 gombot különböztet meg, valamint a gombok lenyomása és felengedése (kattintás), ennek ismétlése (dupla vagy tripla kattintás), illetve a gomb nyomva tartása műveleteket kezeli) és a billentyűzet szabványos kezeléséhez szükséges kódtáblákat.

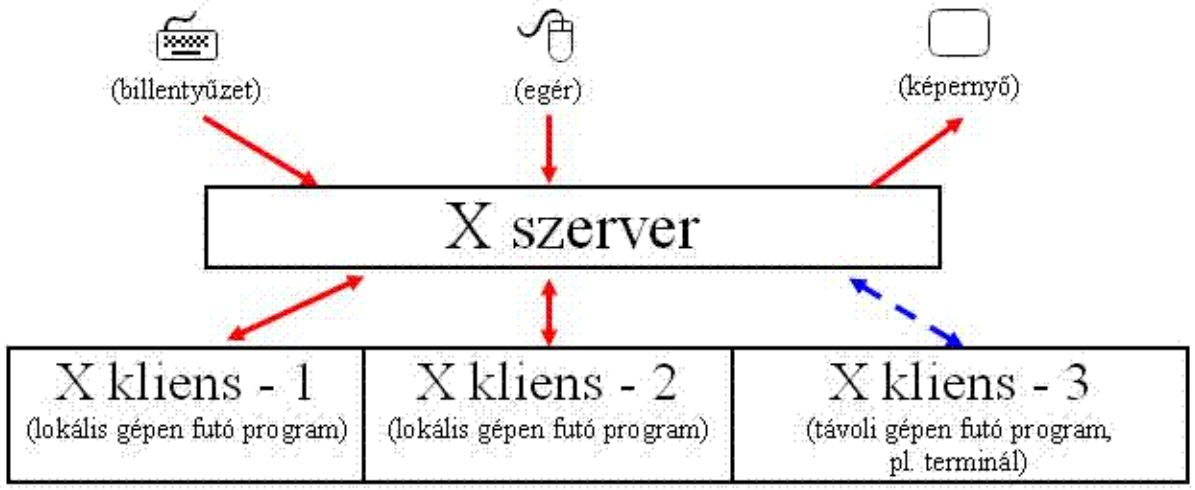
Az ablakkezelők ezeket az alapfunkciókat egészítik ki rendszerint részint felületi (pl. egy ablakot elláthat kerettel, egységes szerkezetet biztosít), részint működési (birtokolhatja a képernyő egészét) jellemzőkkel. Ablakkezelőből csak egy lehet egyszerre a képernyőn, de egy X szerverhez bármennyi X kliens csatlakozhat. (Ismertebb ablakkezelők: MS Windows, KDE/Gnome, Mac OS X).

A X11 működési filozófiája (köszönhetően annak, hogy eredendően hálózati környezetre tervezték) az esemény-vezérlés: a szerver és a kliens üzenetek formájában kommunikál, az egyes üzenetek hatására pedig megváltozik a rendszer állapota. Az üzenetek csomagok formájában közlekednek a két fél között, az egyes csomagok a következő állapotokat hordozhatják:

- **kérés**: a kliens küldi a szervernek, tartalma lehet pl. állapot lekérdezése vagy egy tevékenység elvégzésére irányuló kérés
- **válasz**: a szerver visszajelzése (reakciója) egy kérésre
- **esemény**: a szerver küldi a kliensnek, egy adott tevékenység leírása (pl. a felhasználó lenyomta az egér egyik gombját, vagy egy ablaknak megváltozott a mérete)
- **hibaüzenet**: a szerver küldi a kliensnek, az érvénytelen kérésekre vonatkozó speciális válasz.

Esemény-vezérelt működési alapelve, hogy az operációs rendszer a felhasználó tevékenységét (és a rendszer működésében megjelenő állapotváltozásokat is) folyamatosan figyeli – ez természetesen nem csak a GUI jellemzője, de ez esetben a szokásos műveletek kiegészítéséről van szó. Amennyiben a rendszer környezetében valamilyen változás áll be (a fenti tevékenységek valamelyikének hatására) – ezt nevezzük eseménynek –, akkor erről a grafikus felület szerver-szolgáltatása értesítést küld az operációs rendszernek – ez az üzenet –, a grafikus felület kliense pedig az üzenet információi alapján meghatározza az elvégzendő műveletet. (Tulajdonképpen a karakteres felület parancsértelmező komponensének a megfelelőjéről van szó.) Az elv – látszólag – egyszerűsége ellenére is tökéletes, ez azonban nem feltétlen igaz: a grafikus felületű rendszerek (nem kötelező jelleggel, de leggyakrabban) multiprogramozottak is – ekkor viszont felmerül a kérdés, hogy egy esemény megjelenésekor hogyan azonosítható a forrása?

Amennyiben az esemény egy pozícionáló eszköztől származik (pl. az egér egyik gombjának lenyomása), akkor az eszköz pozíciójának ismeretében meghatározható a kérdéses program. Ha az esemény billentyűzetről származik, akkor az operációs rendszer alapértelmez: a futó programok között mindig van egy aktuális (általában a legutoljára kiválasztott), és ha az eseményből a forrás nem meghatározható, akkor az az esemény az aktuális programra vonatkozik.



2-6. ábra: X Window kliens-szerver architektúra

A GUI komponensei

Habár a grafikus felületű rendszerek felületükben nem feltétlen, de működésükben általában mutatnak rokonságot: lényegében az X11 eszközkészletét valósítják meg illetve (egyes esetekben) egészítik ki. A szabványos komponensek angol nevéből képzett mozaikszóval ezeket a rendszereket szokás WIMP elvű rendszereknek is nevezni: ablak (window), ikon (icon), menü (menu) és pozícionáló eszköz (pointing device) együtteséről van

szó. Az X11-nél már említettük, hogy a szabvány nem sokat törődik a konkrét megvalósításokba implementált megoldásokkal (emlékezzünk: ez az ablakkezelő feladata!), ezért a következőkben a felhasználói felületek gyakorlati szempontból jellemző komponenseit tekintjük át:

Ablak: minden feladathoz hozzárendelhető a képernyő egy (alapértelmezés szerint téglalap alakú) területe: ez a terület a program futási környezete (a grafikus felület szempontjából), az ablak. Multiprogramozott rendszer esetében egy időben tetszőleges számú ablak lehet a képernyőn, de a felhasználó egy időben csak eggyel kommunikálhat (aktív ablak, általában az utoljára kiválasztott). Az ablakok alapértelmezés szerint a képernyő tetszőleges tartományán elhelyezkedhetnek (megengedve a két végletet: a képernyő teljes terjedelmét, illetve a képernyő-területet nem foglaló állapotot és a kettő közti összes lehetséges átmenetet), akár egymást átfedő módon is. Az ablakok alapvetően három csoportba sorolhatók:

- *csoportablak*: azonos típusú objektumok (általában ikonok, ld. később) rendezett megjelenítésére szolgál;
- *alkalmazás-ablak*: az adott alkalmazás (program) munkaterülete,
- *párbeszéd-ablak*: a felhasználói interakció eszköze.

Ikon: kis méretű grafikus ábra, amely egy objektum vagy tevékenység reprezentálására szolgál. Az ikonok az általuk megjelenített objektum tartalma és viselkedése szerint lehetnek:

- *program(indító)-ikonok* (parancsikonok), amelyek valamilyen tevékenységet vagy feladatot, azaz egy konkrét számítógépes programot reprezentálnak,
- *állományszervezési ikonok* (mappaikonok, állomány-ikonok), amelyek a tárolási rendszer különböző szintjeit ábrázolják,
- *hivatkozás-ikonok*, amelyek a tényleges (fizikai) elhelyezkedéstől független (logikai) csoportosítást tesznek lehetővé,
- *alkalmazás-ikonok*, amelyek a képernyőn meg nem jelenő ablakokat jelképezik.

Menü: (az operációs rendszer vagy egy adott program) tevékenységek szöveges megjelenésű – de grafikus viselkedési jellemzőkkel is rendelkező –, strukturált rendszere. A menüszerkezetben menüpontok (csoportok) és menüparancsok (utasítások) szerepelnek. A „grafikus viselkedés” kitétel azt jelenti, hogy ezek a szöveges objektumok (az eseményvezérlő működésének megfelelően) képesek megváltoztatni a megjelenési módjukat (a leggyakrabban alkalmazott ilyen változtatás a parancs kiválaszthatóságának színnel történő jelzése). A menüpontok között vannak szabványos (minden alkalmazásra jellemző, többé-kevésbé azonos menüparancsokat tartalmazó) elemek, ilyenek a Fájl menü („File”, jellemzően az állománykezeléssel, nyomtatással kapcsolatos parancsokkal), a Szerkesztés menü („Edit”, kijelölés, vágólap, keresés parancsokkal) és a Súgó. A menük működésük szempontjából két félék lehetnek: legördülő („*pull-down*”) és felugró (helyi menü, gyorsmenü, „*pop-up*”) típusúak.

A **pozícionáló eszközök** tekintetében általánosságokat meglehetősen nehéz megfogalmazni, hiszen ezen eszközök családja meglehetősen népes: ide tartoznak a különböző egerek, gördítő-csúszó kijelölők (touch-ball, touch-pad), érintő (marker) – eszközök (fényceruza, érintőképernyő), stb.

És végül két megjegyzés:

- egyrészt legyen bármilyen jól szervezett egy grafikus felhasználói felület, a felhasználói interakció nem feltétlen merül ki az előre definiált menük és ikonok

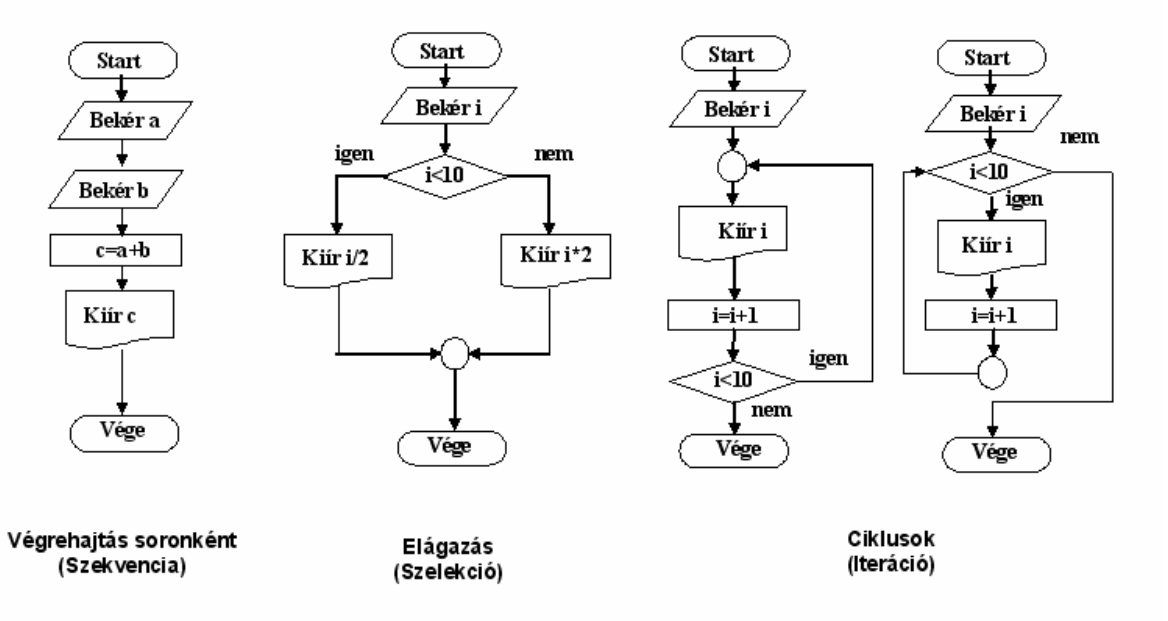
használatában, ezért a GUI-k ablakkezelői az egyes ablakokat további (a felhasználó döntési lehetőségeit kiterjesztő vagy leegyszerűsítő) ún. vezérlőelemekkel ruházzák fel (mint pl. beviteli mezők, parancsgombok, listák, kiválasztó vezérlők: rádiógombok, jelölőnégyzet), stb.) – ezek az eszközök a szabvány kvázi kiegészítésének tekinthetők;

- másrészt, hogy egy grafikus felhasználói felület sem nélkülözheti teljes egészében a karakteres vezérlés lehetőségét: az egyes (általában gyakran használt) tevékenységekhez billentyűkódok épp úgy hozzárendelhetők (és általában ilyen hozzárendelések léteznek is, sőt bizonyos esetekben a felhasználó igényeinek megfelelően módosíthatók, kiegészíthetők), mint ikonok, illetve a WIMP komponensek közül legalább a menü kezelhető és a pozícionáló eszköz helyettesíthető (szimulálható) a billentyűzetről.

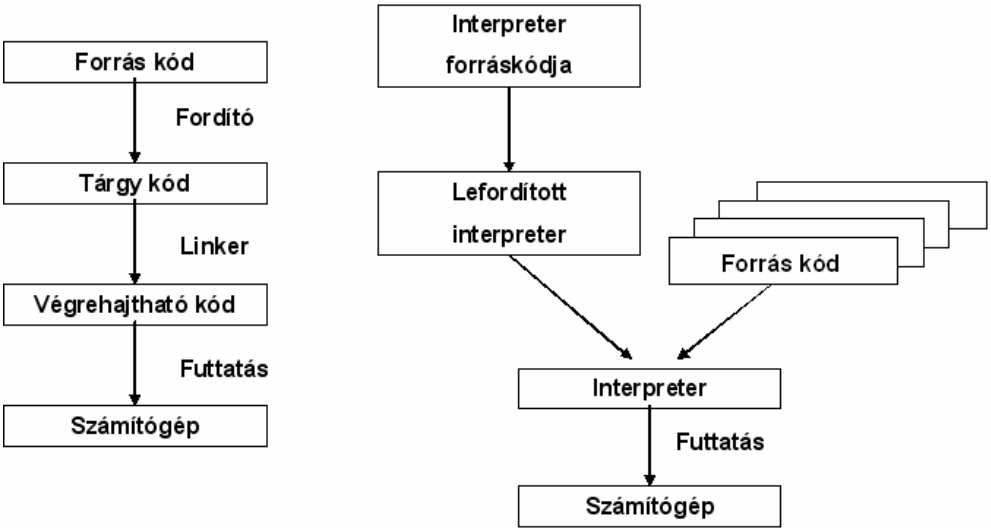
2.3.2. Programok végrehajtása

Ahhoz, hogy megismerjük hogyan futtat a számítógép a kernel segítségével egy felhasználói, vagy egy, a burok (shell) részét képező programot a következőket ismertük már meg. Nevezett programok részére a kernel egy virtuális gépet biztosít, mely segítségével a program a háttértár és az operatív tár között adatokat tud mozgatni, a bemeneti egységekről adatokat tud beolvasni, és a kiíró egységen a számítási eredményeket rögzíteni tudja. Megismertük, hogy a kernel funkciók segítségével ez a virtuális gép hogyan működik, milyen stratégiák segítségével tesz eleget feladatának.

Bár a programokat sokféle logika mentén készíthetünk, automata nyelvek, logikai nyelvek, moduláris programozás, objektum orientált nyelvek, valósidejű és/vagy párhuzamos programozásra alkalmas nyelvek, egy biztos, ha fordítás után a Neumann-elvű gépen szeretnénk futtatni a programot, a virtuális gép részére érthető utasításokra kell fordítani ezeket, azaz, röviden, gépi kódra. A gépi kódra fordításnak két fontos összetevője van, az egyik a változó, a hozzátartozó változó típussal, ami megmondja, hogy kell a változó értékét tárolható bitsorozattá konvertálni, memóriacímmel, ami megmondja, hogy az operativ tárban honnan kezdődik az előbb említett bitsorozat, és meddig tart, valamint értékkel, aminek tárolására eleve létrejött. A másik a programozási struktúrák *(2-7. ábra)*. Természetesen ezek a struktúrák gépi kód esetén még apróbb építőelemekből épülnek fel, címke, ugró utasítás, ill. feltételes ugró utasítás.



2-7. ábra: Programozási szerkezetek



2-8. ábra: Programozási nyelvek működése[13]

Az felhasználói programok írására használt magasszintű programozási nyelvek két típusba sorolhatók, a fordító (compiler) nyelvek illetve az értelmező (interpreter) nyelvek. *(2-8. ábra, 2-1. táblázat).*

[13] http://www.isg.rhul.ac.uk/msc/teaching/iy5512/wk03/comp_systems_4up.pdf alapján

2-1. Táblázat: A programok működése

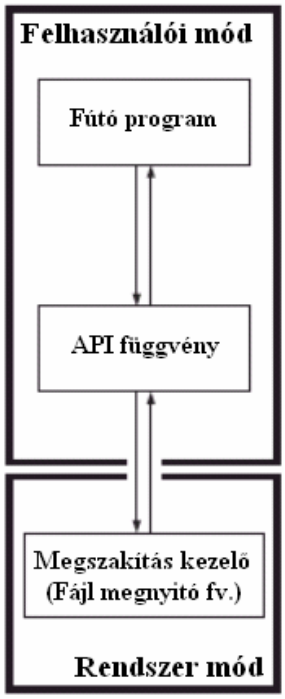
A fordító programok működése	Az értelmező programok működése
1. A forráskód írása • Valamilyen magas szintű programozási nyelven pl. Pascal, C# • Ez még nem valódi végrehajtható kód 2. A forráskód tárgy kódra fordítása • Gépikódú utasítások létrehozása, memóriacímek meghatározása 3. Tárgykódok összefűzése 4. A tárgykód betöltése a memóriába 5. A program futtatása az operációs rendszer által szolgáltatott virtuális gépen	1. A forráskód írása • Valamilyen funkcionális nyelven vagy logikai nyelven (Prolog) 2. A program végrehajtása az értelmező segítségével, lehetséges változatok: • Az értelmező operációs rendszer nélkül is elérhette a hardvert, pl. BASIC • Az értelmező az operációs rendszer felett is futhat, pl. PERL, Prolog • Saját virtuális gépén keresztül éri el az operációs rendszer virtuális gépét, pl. JAVA, .NET

A számítógépes programok a programozási szerkezetek egymásba ágyazásából felépített kisebb egységekből, úgynevezett blokkokból épülnek. Egyes blokkok jól meghatározott feladatot látnak el, és a program futása során többször meghívhatók. Ezeket a többször meghívható (felhasználható) egységeket függvénynek, eljárásnak, modulnak, vagy osztálynak nevezzük. Az egyes blokkok egymással kommunikálni tudnak, a változók értékeit, címeit egyes esetekben egymásnak átadhatják, ezt a lehetőséget paraméterátadásnak nevezzük. Azok a blokkok melyek meghívásuk után az általuk számított eredményt értékadással egy változónak átadhatják, függvénynek, míg azokat a blokkokat melyek csak a változók értékeinek megváltoztatására jöttek létre, eljárásnak nevezzük.

Léteznek speciális, megosztott függvénykönyvtárak (shared libraries) melyek előre megírt függvényeket bocsátanak a felhasználói programok íróinak részére. Például szövegkezelő, matematikai, kriptográfiai, sőt (operációs) rendszer függvények. Ezeknek a dll (Dynamic Link Library) fájloknak egy részét az operációs rendszerek telepítéskor az alkalmazások és leendő fejlesztők részére bocsátják, másik részükhöz az alkalmazások telepítésével, esetleg egy fejlesztői környezet beszerzése és telepítése után juthatunk. A megosztott függvénykönyvtárak mint típus elnevezésére, absztrakt definíciójára használjuk a felhasználói program interfész („*application program interface*”, röviden API) kifejezést. Az API feladata, hogy típusa megvalósítása esetén, azaz implementációja révén, közvetítsen a felhasználói programok, és a rendszermódban futó programok között, azaz lehetővé tegye a rendszerhívásokat. A különböző operációs rendszerekhez általában eltérő API-k állnak rendelkezésre, az ismertebbek ezek közül

- a Win32 API, amely a Windows rendszereket szolgálja ki ;
- a POSIX API a Posix kompatibilis operációs rendszerekhez (pl. Unix, Linux, Mac OS X) illeszkedik;
- a JAVA API pedig a Java virtuális gépet[14] szolgálja ki.

[14] Lásd a következő fejezetben



1. A felhasználói program elérkezik ahhoz, hogy egy API fügvénnyel megnyisson egy állományt
2. Az API függvény kicseréli és elmenti megszakítási vektor címét a regiszterben
3. Az API függvény végrehajtja a rendszerhívást, azaz felhasználói módból kernel módba kapcsol
4. Az operációs rendszer átirányítja a megszakításkezelőt a megszakításvektornak megfelelően
5. A megszakításkezelő végrehajtja az aktuális feladatot, visszacseréli a megszakítási vektort, majd a függvény visszatérési értékét átadja az API függvénynek
6. Az API függvény átadja a visszatérési értékét a felhasználói programnak

2-9. ábra: A rendszerhívás folyamata, példa

2.4. Virtuális gépek

Eddigi ismereteink szerint megtudtuk, hogy az operációs rendszer kiterjesztett gép, azaz virtuális gép, elválasztandó a felhasználói programokat a közvetlen rendszereszköz használattól. Ezzel a módszerrel fokozni lehetett a hardver biztonságos működését, lehetővé vált több folyamat látszólagos egymás melletti futtatása (multitask), és egyben könnyebbé lett a magas szintű programozási nyelvek használata.

Léteznek olyan alkalmazások, amelyek lehetővé teszik egy számítógép, vagy egy operációs rendszer emulálását, azaz az operációs rendszer felett létrehoznak egy második virtuális gépet. Ez a második virtuális gép emulálja a CPU-t, a Be/Ki meneti eszközöket, sőt, saját BIOS-szal is rendelkezik. Természetesen az eredeti számítógép teljesítményétől függ az emulált eszközön futó program teljesítménye, de működni, működik. Például Mac-on-Linux, Bochs, CoLinux, MS Virtual PC, Vmware, stb.

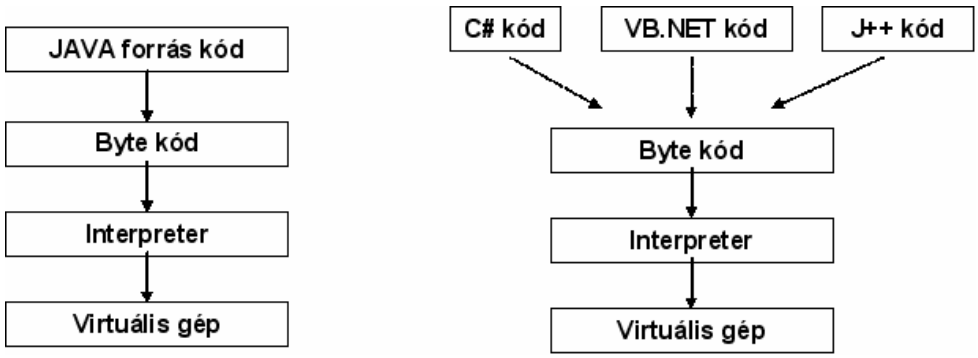
A Java virtuális gépet (JVM) a Sun Microsystems kezdte kifejleszteni 1995-ben. Célja egy platform-független, hordozható („*write once, run everywhere*”: „írd meg egyszer, használd bárhol”) alkalmazásokat létrehozó fordító kifejlesztése volt. Erre azért van szükség, mert ha egy adott nyelven, egy adott operációs rendszeren megírunk egy programot, akkor más operációs rendszeren való futtatásához előbb újra kell fordítanunk, esetleg le kell gyártanunk a megfelelő API-kat, azaz portolnunk kell, és ez rendkívül munkaigényes. A platformfüggetlenség úgy lehetséges, hogy a tetszőleges operációs rendszerhez rendelkezésre kell hogy álljon egy felette futó virtuális gép is, jelen esetben a JVM. A JAVA[15] fordító bájtkódokból álló programsorokat állít elő, melyet a JVM-et felügyelő értelmező (interpreter) hajt végre. A Java az Internet nyelve, és a különféle gépeken, különböző operációs rendszerek alatt megjelenített weboldalakhoz kapcsolódó kliens oldali Java programoknak eltérő környezetekben kell futniuk. Természetesen a hordozhatóságért cserébe a JAVA programok futtatásának erőforrásigénye nagyobb, mint az adott operációs rendszerhez gyártott

[15] A JAVA objektum elvű, imperatív nyelv, a C/C++ hoz hasonló szintaxissal

programnyelvvel létrehozott alkalmazások erőforrásigénye, amit a felhasználó egy rosszul megválasztott hardveren lassúságként él meg.

.NET project 1998-ban kezdődött, mikor a Microsoft J++ projekt befejeződött. Célja a Windows API-kat felhasználó VM létrehozása, alapja az OmniVM volt. .NET keretrendszer segítségével a Microsoft szerver alkalmazások, fejlesztőeszközök portolás nélkül, csak a megfelelő virtuális gép telepítésével hordozhatóvá válnak.

A JVM és a .NET keretrendszer segítségével nem hagyományos számítógépek például okostelefonok, zseb PC-k, PDA-k is bevonhatók a számítástechnika, Internet világába, így e két virtuális gép versenye tág teret nyithat a programozható eszközök bővülésének (*2-10. ábra*)



2-10. ábra: A JAVA technológia és a .NET keretrendszer

2.5. Állományszervezés

Az operációs rendszereknek az állományszervezéssel és a háttértárak kezelésével kapcsolatos tevékenységei (felhasználói szempontból legalábbis) majdnem olyan jelentősek, mint a kommunikációt megvalósító szolgáltatások. A következőkben röviden áttekintjük a (gyakorlati szempontból) jelentősebb háttértárak esetében alkalmazott jellemző működési, tárolási, és kezelésével kapcsolatos alapvető elveket, módszereket.

2.5.1. Háttértárak

Azokat a kétirányú perifériákat, amelyek képesek az információ tartós tárolására, háttértáraknak nevezzük. A háttértárak jelentőségét a számítógép működésbeli sajátosságai közül az operatív tárnak (memóriának) azon jellemző tulajdonsága adja, mely szerint az ilyen memória tartalma a számítógép kikapcsolását követően elvész. A háttértárak esetében alkalmazott tárolási elvek a számítástechnika fejlődésével számos változáson mentek keresztül (mechanikus elvű háttértárak: lyukkártya, lyukszalag, mágneses adathordozók: mágnesszalag, mágneslemez, optikai eszközök: magneto-optikai lemez, CD, DVD, elektronikus elvű tárak: pen-drive). Az egyes háttértárak jellemzésére olyan tulajdonságok szolgálnak, mint a kapacitás, a jellemző (átlagos vagy maximális) adatátviteli sebesség, az elérési idő vagy a megbízhatóság. A háttértárak tárolási kapacitása lényegesen nagyobb, mint az operatív táré – azonban a háttértáron tárolt információ elérési ideje is lényegesen nagyobb (azaz a háttértár tartalmával való műveletvégzés lassítja a rendszer működését): ennek a kettősségnek a kezelése az operációs rendszer alapfeladata.

2.5.1.1. Alapelvek

A háttértáron elhelyezkedő információ feldolgozásához alapvetően két tevékenységet kell az operációs rendszernek elvégeznie: megtalálni a háttértáron a kérdéses adatot tartalmazó részt (elérési idő), majd elvégezni a kérdéses adatmennyiség mozgatását a háttértár és a memória között (átviteli idő). (Ezen (alap)tevékenységek megvalósításában az egyes háttértárak természetesen jelentős eltéréseket mutatnak (pl. a háttértár típusának, működési

elvének függvényében), de általánosságban minden háttértárra jellemzők, így alkalmasak arra, hogy rajtuk keresztül vizsgáljuk meg az operációs rendszerek ezen feladatok kezelésével kapcsolatos tevékenységeit.)

Nyilvánvaló, hogy amennyiben ezeknek az időtartamoknak valamelyikét csökkentjük, úgy a rendszerünk működése gyorsabbá válik. Az átviteli idő elsődlegesen egy technikai paraméter, abban az értelemben legalábbis, hogy nagyságát alapvetően az adott háttértár működési mechanizmusa és a számítógép hardver architektúrája: az alkalmazott sínrendszer határozza meg. Érdemi változtatására az operációs rendszernek általában nincs lehetősége – ezzel együtt léteznek olyan operációs rendszerek, amelyek ezzel is megpróbálkoznak, a legtriviálisabb módszer a röptömörítés alkalmazása. Az elérési idő ezzel szemben egy jól szervezhető jellemző, alkalmas megoldásokkal nagyságrendi változtatások érhetőek el a mértékében.

A **röptömörítés** (ahogy azt a neve is sugallja) egyfajta tömörítési algoritmus alkalmazását jelenti, a „röp” jelző a működés azon jellemzőjére utal, hogy nem szükséges felhasználói tevékenység a végrehajtásához, az állomány kezelésekor automatikusan megtörténik („on-the-fly”). Amennyiben az operációs rendszerünk támogatja a háttértár ilyen módon történő kezelését, akkor az állományok háttértárra történő mentésekor az operációs rendszer előbb betömöríti a kérdéses állományt, majd az így kapott archívumot helyezi el a háttértáron, beolvasáskor pedig egy automatikus kitömörítési művelet hajtódik végre. A röptömörítés alkalmazása esetén az átviteli idő csökken (kevesebb információ mozog a memória és a háttértár között), azonban az állomány feldolgozásának az ideje megnő (az írás/olvasás művelet mellett egy újabb – és számítás-igényesebb, tehát lassabb! – művelet elvégzésére is szükség van). (A röptömörítés használata a támogató operációs rendszerekben általában logikai állományszervezési egységenként (állomány, könyvtár) adható meg.)

Az elérési idő csökkentésének egy viszonylag gyakori (és az adott háttértár működési elvétől független) módja a gyorsítótárak (cache) alkalmazása.

A **gyorsítótárak** (megjegyezzük, hogy nem csak a háttértáraknál alkalmazott módszerről van szó!) működése az ún. „*lokalitás-elv*”-en alapszik: megfigyelés (de statisztikailag is igazolható), hogy *ha egy adatra szükség van egy számítógépes feldolgozási folyamatban, akkor várhatóan hamarosan szükség lesz az előbbi adat környezetében* („mellett”) *elhelyezkedő adatokra,* információkra *is*. Ebből következik, hogy a (lassú!, ld. fentebb!) háttértárak használata esetén az operációs rendszer nem csak azt az adatot olvassa be, amire a felhasználó kérése vonatkozik, hanem még néhány mellette lévőt is. Az így beolvasott adathalmazt tárolja (átmenetileg) a gyorsítótár. Amennyiben a következő művelet (ugyanarra az eszközre vonatkozóan) – az ilyen módon az előbbiek során már beolvasott valamelyik adatra vonatkozik, akkor azt nem kell még egyszer behozni (már rendelkezésre áll).

2.5.1.2. Mágnesszalag

A mágnesszalag („tape”, DAT) nem tartozik a legkorszerűbb háttértárak közé, azonban archív tárként történő alkalmazása mind a mai napig jelentős. Mágneses elvű háttértár, az adatok tárolása a mágneses indukció elvén alapszik (egy műanyag hordozóréteg felületére olyan anyagot viszünk fel, amely elektromos áram hatására megváltoztatja mágnesezettségi jellemzőjét, a szalag fölött rögzített pozícióban helyezkedik el az író-olvasó fej). A mágnesszalag (viszonylag) nagy kapacitású (2-40 GB), soros elérésű (egy adott adat eléréséhez végig kell olvasni az összes megelőző adatot), 2-10 Mbit/s átviteli sebességű (azonban a soros elérésből adódóan az elérési idő (pozícionálás) viszonylag nagy mértéke miatt ez felhasználói szinten nem ilyen értékben jelenik meg) háttértár. A mágnesszalag tárolási szempontból blokk-szervezésű: az egyes állományok tartalma azonos (ritkábban

változó) méretű részekben (rekord) kerül felírásra, és mind az egyes blokkokat, mind az egyes állományokat üres helyek („gap”) választják el egymástól. (Ennek a jelentősége abban áll, hogy a szalag elején elhelyezkedő „tartalomjegyzékben” megjelölve azt, hogy a kérdéses állomány hanyadik a szalagon, az elérési idő nagymértékben csökkenthető oly módon, hogy az író-olvasó fej gyorscsévélés közben is képes érzékelni ezeket az üres helyeket.)

2.5.1.3. Mágneslemez

A mágneslemezes háttértárakat (tárolási elvük megegyezik a mágnesszalagos háttértáraknál leírtakkal azzal a különbséggel, hogy az író-olvasó fej nem rögzített, ld. lentebb) a tároló adathordozó jellemző fizikai tulajdonsága alapján két csoportba sorolhatjuk: hajlékony-lemezes (floppy disk, FDD) és merevlemezes (winchester, HDD) háttértárakra. Mindkét típus közvetlen elérésű (a háttértár egyes tárolási egységeinek az elérési ideje kb. azonosnak tekinthető, bármelyik tárolási hely tartalma hozzáférhető az őt megelőző részek feldolgozása nélkül), blokk-szervezésű (bár a blokkok mérete (a háttértár méretétől és az alkalmazott állományszervezési módszertől függően bizonyos határok között) eltérhet, de mindenképpen 512 bájt 2-hatványú többszöröse). A FDD-kre 360 KB és 1,4 MB közötti, a HDD-kre (napjainkban) 10 GB és 200 GB közötti tárolókapacitás és 5-50 MB/s átviteli sebesség jellemző. (A mágneslemezes háttértárak esetében alkalmazott jellemző még a fordulatszám (RPM: fordulat/perc), értéke 360 (FDD) és 12000 (HDD) között változhat.)

A mágneslemezes háttértárak tárolási rendszere sáv-szektor szervezésű: a lemez felületén koncentrikus körök (sávok) mentén történik az adattárolás, a tárolás egysége a sáv egy körcikke (szektor). A lemezfelület fölé „nyúl be” az író-olvasó fej, ami sáv-irányú elmozdulásra képes, a szektorok elérhetőségét pedig a lemez forgása biztosítja. A hajlékony és a merevlemez között (legalábbis a működés vonatkozásában) az az alapvető különbség, hogy a merevlemezes háttértárban több lemez kerül egymás fölött (közös tengelyen) elhelyezésre – természetesen ebben az esetben minden lemezfelülethez külön író-olvasó fej tartozik, azonban azt meg kell jegyeznünk, hogy ezek a fejek (a lemezekhez hasonlóan) közös tengelyen rögzítettek. A különböző lemezeken egymás fölött (azonos helyen) elhelyezkedő sávokat cilindernek nevezzük. Az egyes tárolási egységeket hagyományosan sorszám azonosítja (a sávok és a szektorok számozása általában 0-val, a cilinderek (vagy ami ezzel egyenértékű, az író-olvasó fejek) számozása 1-gyel kezdődik).

A fentiek alapján viszonylag könnyű belátni, hogy egy mágneslemezes háttértáron elhelyezkedő adat eléréséhez három információra van szükség: melyik lemez hanyadik sávjának hanyadik szektora tartalmazza a keresett adatot, ezt a hármast nevezik (az angol terminológia kezdőbetűiből) CHS-nek nevezik (Cylinder (=sáv), Head (=az író-olvasó fej), Sector (=szektor)).

A fentiekből egy érdekes következtetés adódik: a mágneslemezek forgási sebessége állandó (ld. RPM), ekkor viszont a lemez külső széléhez közelebbi sávokat alkotó szektorok lényegesen tovább tartózkodnak az író-olvasó fej zónájában, mint a belsőbb sávokon elhelyezkedők. Azonos átviteli sebesség mellett ez azzal jár, hogy a külső sávokon

- a szektorokat nem használjuk ki maradéktalanul (eltérő adatsűrűségű szektorok), vagy
- több szektort hozunk létre, mint a belső sávokon.

A mágneslemezes háttértárakon a fenti tárolási szerkezet kialakítására szolgáló műveletet nevezzük formázásnak (formatálás).

2.5.1.4. Optikai háttértárak

Az optikai elvű háttértárak esetében az információtárolás a fényvisszaverődés elvén alapul: a hordozó felületen olyan elrendezést alakítanak ki, amelyet nagy energiájú fénysugárral megvilágítva a visszaverődés ideje eltérő lesz a tárolt információ 1 illetve 0 értékének megfelelően. Az eltérést eredetileg a felület egyenetlensége biztosította: a hordozó rétegen sík részek („land") és bemélyedések („pit") váltakoztak (a bemélyedésekből természetesen hosszabb idő alatt verődik vissza a fény), azonban az ilyen elrendezés kialakítása speciális eszközöket igényel (préseléssel gyakorlatilag csak a gyári körülmények között előállított CD-k („Compact Disk") esetében találkozhatunk). Az „otthoni körülmények között" írható CD-k esetében a fényvisszaverődés időeltolását a hordozó rétegben eltérő sűrűségű részek kialakításával oldják meg (az olvasáshoz képest eltérő energiájú lézerrel megvilágítva a felületet, azon az adattároló réteg megsűrűsödik). (Az újraírható CD-k esetében ez még annyival bonyolódik, hogy egy harmadik energiatartományba eső fénysugárral az előzőleg felírt (tehát anyagában sűrűbb részeket) visszaalakítják – ez a törlés művelete.) A DVD-k („Digital Video/Versatile Disk") esetében az alapelv ugyanez, csak az alkalmazott technológiának köszönhetően lényegesen nagyobb pontosság érhető el, továbbá a DVD lemezek esetében akár mindkét oldalon kialakítható (akár egymás fölött két-két) adathordozó réteg.

A CD-k esetében a tárolt tartalom kezelése szempontjából alapvetően három típust kell megkülönböztetnünk: a CD-ROM („Read Only Memory") csak olvasható, a CD-R („Recordable") felhasználói eszközökkel írható (de nem törölhető) és a CD-RW („Rewritable"), ami (ugyancsak felhasználói eszközökkel) többször is írható (azaz írható és törölhető. A DVD-k esetében annyival összetettebb a kép, hogy jelenleg két szabvány is létezik (az egyiket „+", a másikat „–" jel jelzi), valamint a fenti típusok (ROM, R, RW) mellett léteznek DVD-RAM lemezek is.)

Az optikai háttértárak közvetlen elérésűek, blokk-szervezésűek (az adattárolás a lemez közepéről induló, kifelé bővülő sugarú spirál mentén, adatblokkok formájában történik – azonban az egyes blokkok szerkezete lényegesen eltér a mágneses elvű háttértáraknál alkalmazottaktól, és az adattárolás alapegysége sem a blokk, hanem a blokkok csoportja, a szektor), szabványos átviteli sebességük 150 KB/s. (Az egyes lemezeken, illetve meghajtókon látható nX jelölések (pl. 24X, 48X) azt fejezik ki, hogy az adott eszköz a szabvány átviteli sebesség hányszorosával képes kommunikálni.) A kapacitás tekintetében szintén elég széles a skála: a CD-k esetében (szabványosan) 650 MB (bár léteznek 700 MB, illetve 800 MB-os lemezek is), a DVD-k esetében 4,5 GB (egyoldalas, egyrétegű) – 19 GB (kétoldalas, kétrétegű).

Az optikai háttértárak működési jellegzetessége az állandó kerületi sebesség (szemben a mágneslemezek állandó forgási sebességével), ami azt jelenti, hogy a lemez minden része azonos ideig tartózkodik a lézersugár letapogatási zónájában – ez viszont csak úgy lehetséges, ha a lemez forgási sebessége az elérni kívánt adat helye szerint változik: a belső részeken elhelyezkedő adatok esetében lassul, a külső szélek felé haladva egyre gyorsabb. (Ez a folyamatos sebesség-változtatás (felpörget-lelassít) adja a CD-k jellemző „búgó" hangját.)

Az optikai elvű háttértárak esetében a tárolt információ típusának függvényében a tárolási szerkezet (gyakorlatilag az adatblokkok szerkezete) eltérő lehet – az egyes típusokra jellemző szabványokat különböző színnel jelölt szabványgyűjtemények („Rainbow Books" – kb. „szivárvány-könyvek") tartalmazzák:

- „Red Book" („Vörös"): CD-DA (audió CD), az alapszabvány;
- „Yellow Book" („Sárga"): CD-ROM

- „Orange Book” („Narancssárga”): CD-ROM/XA, kiterjesztett módú CD: eredetileg képek tárolásához dolgozták ki (Photo CD), de gyakorlatilag az írható-újraírható optikai adathordozók (CD-R, CD-RW) szabványává vált;
- „White Book” („Fehér”): CD-ROM/XA mode 2: videó CD-k szabványos formátuma, stb.

2.5.1.5. Elektronikus elvű hordozható tárak

Az elektronikus elven működő háttértárak a tároló típusú perifériák legújabb generációját képviselik. A különböző típusú memória-kártyák és a (tárolási képesség mellett általában további szolgáltatásokkal is felruházott) pen-drive-ok ugyan hordozható tárolóeszközök, de működési jellemzőik tekintetében sokkal közelebb állnak a memóriákhoz, mint a háttértárakhoz. Az alapelv a memória-kezelésben EPROM-ként (elektronikusan programozható ROM) és az EEPROM-ként (elektronikusan törölhető programozható ROM) ismert módszerek továbbfejlesztése: egy ROM típusú (tehát nem felejtő) memória tartalmának feszültség-értékek felhasználásával történő módosítása. Érdekességük, hogy saját processzorral rendelkeznek, amely az egyébként memória-szervezésű tárat blokk-szervezésűnek (és ilyen módon állomány-elhelyezésre alkalmas formátumúnak) mutatja az operációs rendszer felé.

2.5.2. Fizikai állományszervezés

Az állományszervezés az operációs rendszer azon feladatainak gyűjteménye, amelyek biztosítják a felhasználó által adott információ háttértáron történő elhelyezését, illetve visszakeresését. Ez a tevékenység alapvetően két szinten valósul meg:

- egyrészt a felhasználó által „érzékelt” szinten (logikai szint): itt állományokkal, könyvtárakkal, kötetekkel (meghajtókkal) találkozik a felhasználó,
- másrészt az adott háttértár által „értelmezhető” szinten (fizikai szint). itt a háttértár tárolási rendszerének megfelelő fogalmakkal (CHS, blokk) dolgozik maga az adott eszköz.

A két szint közötti megfeleltetés kialakítása az operációs rendszer feladata. Ahhoz, hogy ennek a szolgáltatás-csoportnak a működését megérthessük, nézzük meg mindkét szint alapfogalmait.

2.5.2.1. Mágneslemezek

Az előbb a mágneslemezek esetében azt mondtuk, hogy az adattárolás szervezése szektor alapú (1 szektorban 512 bájtnyi adat tárolható), a tárolási hely (szektor) meghatározása pedig egy három dimenziós azonosító (CHS) segítségével történik, ezek a jellemzők sorszámozottak. (A könnyebb érthetőség érdekében tekintsünk el a CHS típusú szektorazonosítástól, és tételezzük fel, hogy minden szektornak önálló sorszáma van – ezt az általánosság megszorítása nélkül megtehetjük, sőt még az is igaz, hogy a konkrét megvalósításokban az alábbiakban ismertetettnél szűkebb értéktartománnyal (pl. ez IDE esetében 24 bit) dolgoznak az operációs rendszerek.)

Tegyük fel, hogy az operációs rendszerünk 16 bites címeket használ a szektorok azonosítására. Ebben az esetben 2^16 * 512 bájtnyi helyünk van, ami pontosan 2^16 * 2^9 = 2^25=2^5 * 2^20 = 32 M – azaz 16 bites operációs rendszerben legfeljebb 32 MB méretű háttértárakkal lehet dolgozni (32 bites rendszerben ez a szám 2^32 * 2^9 = 2T, a 64 bites rendszerek esetében a számítást az olvasóra bízzuk). Mivel a címtartomány nem növelhető, a háttértár kapacitásának növeléséhez a blokkméretet kell növelni. Éppen ezért az operációs rendszerek a gyakorlatban nem szektorokat, hanem szektor-csoportokat („cluster”, klaszter) kezelnek a tárolás alapegységeként.

Egy mágneslemezes háttértár egymás után elhelyezkedő szektoraiból álló tárolási egységeket klaszteregeknek nevezzük. A klaszter a háttértár legkisebb címezhető egysége.

Egy szektor-csoportba 2-hatvány darab szektor kerülhet összevonásra (2, 4, 8, stb.), így a szektor-csoportok mérete (elvileg) 1-64 KB között változhat (operációs rendszerenként eltérő, hogy ez a méret alapértelmezett (a háttértár méretének függvényében), vagy felhasználói eszközökkel megadható). A szektor-csoportok használata kettős jellegzetességgel bír:

- egyrészt a háttértáron elérhető tárolóterület nő
- másrészt a háttértár kihasználhatósága romlik (mivel a tárterület legkisebb önállóan kezelhető egysége a szektor-csoport, minden állomány egy (vagy több) ilyen egységben helyezhető el, azaz adott esetben egy 100 bájtos állomány is elfoglal 64 KB területet…)

A tárolási egységek (a továbbiakban az egyszerűség kedvéért a blokk kifejezést használjuk, ami a háttértár szervezésétől függően jelenthet szektort vagy klasztert) tehát sorszámozottak. A háttértár kezelésekor alapértelmezés szerint az összes blokk egyetlen tárterületet alkot, de ez nem szükségszerű: a blokkok folytonosan sorszámozott részei egymástól (logikailag) függetlenül kezelhetők.

A háttértár tárolási egységeinek logikai szempontból együtt kezelt, folytonos indexű sorozatát partíciónak nevezzük, az már operációs rendszerenként változó, hogy az adott rendszer hány darab és mekkora méretű partíció kezelésére képes. Az adott háttértáron elérhető partíciók jegyzékét nem az operációs rendszer, hanem maga a háttértár tartalmazza, mégpedig az első logikai tárolási egységben – ennek a neve a partíciós tábla. (A partíciós tábla tartalmaz egy speciális programot is – MBR: Master Boot Record –, ami a partíciós táblában tárolt információ értelmezésére képes, és nem az operációs rendszer, hanem egy BIOS rutin indítja a számítógép bekapcsolását követően.)

2.5.2.2. Optikai háttértárak

Az optikai lemezek tárolási szerkezete erősen kötődik a lemez tartalmához (pontosabban a tartalom létrehozásakor alkalmazott szabványhoz – ld. „szivárvány könyvek”), de alapelvként a következőket mondhatjuk el:

1. az optikai lemezen az adatok tárolása belülről kifelé bővülő sugarú spirálkarok mentén történik, azonban ezen spirálok nem mindegyike tárol felhasználói szempontból adat típusú információt: a forgatótengelyhez közelebbi sávokon ún. bevezető („lead-in”) jellegű adatok találhatók (ilyen pl. a tartalomjegyzék), a lemez legkülső részén pedig a kivezető („lead-out”) adatok kaptak helyet;
2. az optikai elvű háttértárakon az adatok tárolása blokk-szervezésű, de az egyes blokkokban az adatbájtok mellett informatív és ellenőrző bájtok is elhelyezkednek:

CD-DA esetében egy 24 bájtos blokk (192 bit) tárolása közelítőleg 73 bájton (588 bit, ami az alkalmazott kódolás miatt 42 bájtnak felel meg) történik), a blokkok szektorokba szervezettek (a szektor mérete 98 blokk = 2352 bájt) – ebben az esetben (természetesen) állományszervezésről nem beszélhetünk;

CD-ROM is 2352 bájtos szektorokat alkalmaz, ebben az esetben azonban az adatbájtok száma 2048, amit szinkron-, hibaérzékelő- és hibajavító bájtok egészítenek ki (mode 1-ben legalábbis, mode 2 esetén (ld. CD-ROM/XA) a hibajavító bájtok helyét is adattárolásra használják fel).

2.5.2.3. Működés

A fentiek kezelésével kapcsolatban az operációs rendszernek alapvetően két rendszerszintű tevékenységet kell elvégeznie: címszámítást és ütemezési feladatokat.

A címszámítás ezt jelenti, hogy az operációs rendszernek alkalmas módon meg kell adnia a mágneslemez érintett szektorának azonosítóját (emlékezzünk: CHS!), ez azonban közel sem triviális a klaszterezés illetve a sávonként eltérő szektorszám miatt. Éppen ezért az operációs rendszerek virtuálisan kezelik a klasztereket: a rendelkezésre álló címtartományon belül folytonosnak „képzelik el" őket (figyelem: a partícionálás miatt a fenti címtartomány nem feltétlenül a címzési rendszerből adódó maximális tartomány!), és a háttértár eszközvezérlő elektronikájára bízzák a logikai címhez tartozó CHS-index meghatározását. Ennek a módszernek alapvető előnye, hogy az operációs rendszernek nem kell azzal törődnie, hogy az adott háttértáron milyen a szektorok szervezettsége: az eltérő szektorszámból adódó különbséget az eszközvezérlő kezeli.

Az ütemezés jelentőségének megértéséhez emlékezzünk vissza az elérési idővel kapcsolat tett alapvetéseinkre, amelyet azzal kell kiegészítenünk, hogy egy szektor tartalmának feldolgozása a következő három tevékenység sorozataként valósul meg:

1. a tartalmazó sáv kiválasztása (az író-olvasó fej megfelelő sávra történő pozícionálása – elérési idő)
2. a sávon belül a kérdéses szektor kiválasztása (a lemez forgásából adódóan a keresett szektornak az író-olvasó fej alá fordulásának ideje – lappangási idő)
3. a szektor olvasása/írása (átviteli idő).

A három jellemző közül az első nagyságrendekkel nagyobb a második kettőnél (nem szólva arról, hogy az utóbbi kettő adott háttértár esetén kvázi állandónak tekinthető), és mivel egy írási/olvasási műveletben általában különböző sávokon elhelyezkedő szektorok vesznek részt, ezért nem mindegy, hogy az adott művelet elvégzéséhez hányszor kell az író-olvasó fejet pozícionálni (pontosabban hogy az egyes pozíciók között mennyit kell mozgatni). A konkrét megvalósításokban a következő módszerek fordulnak elő:

- **FCFS** (sorrendi kiszolgálás): a legegyszerűbb ütemezési stratégia (tulajdonképpen nem is nevezhető stratégiának, hiszen semmilyen szervezést nem használ): az író-olvasó fej mozgatása az igényelt sávsorrendnek megfelelően történik
- **SSTF** (minimális sávtávolság alapú kiszolgálás): a pozícionálás a műveletben szereplő sávok közül mindig arra a sávra történik, amely a legutoljára elért sávhoz a legközelebb helyezkedik el – gyors, de veszélyes stratégia, hiszen a lemez töredezettségével (ld. később) párhuzamosan ezzel a stratégiával az író-olvasó fej műveletenként ellenkező irányba mozog („rángat"), ami nem tesz jót a mechanikájának…
- **SCAN** (pásztázó) illetve **LOOK** (kereső) stratégiák: szokás őket lift-algoritmusoknak is nevezni, a működési elv a felvonókéhoz hasonló: az író-olvasó fej a mozgás irányába eső kérések mindegyikét kiszolgálja, függetlenül azok érkezésének sorrendjétől (a két módszer között az a különbség, hogy a SCAN a lemez tényleges sávindexei, a LOOK pedig csak az igényelt sorozatban előforduló sávindexek szélsőértékei között mozog).

2.5.3. Logikai állományszervezés

A háttértárak fentebb említett szerkezeti és működésbeli sajátosságaiból a felhasználó (bár aktívan használja ezeket az eszközöket, módszereket) általában nem sokat tapasztal meg

közvetlen módon. A háttértárak kezelése felhasználói szempontból elsődlegesen állományok elhelyezését jelenti - az operációs rendszer szintjén természetesen ezek is az állományszervezéssel kapcsolatos tevékenységek, csak éppen egy másik értelmezésben. Ebben a megközelítésben az állományszervezéssel kapcsolatos műveletek közül azokat, amelyek a háttértárak kezelésével kapcsolatosak (azok működési sajátosságait szem előtt tartva működnek), az állományszervezés fizikai szintjének, azokat, amelyek a felhasználói tevékenységgel kapcsolatosak, logikai szintnek nevezzük. Ebben a részben a logikai állománykezelő működési sajátosságait tekintjük át.

2.5.3.1. Alapfogalmak

Az állományok (fájlok) és könyvtárak (mappák, katalógusok) fogalma valószínűleg minden számítógéppel dolgozó felhasználó számára ismert – de vajon hogyan határoznánk meg a jelentésüket?

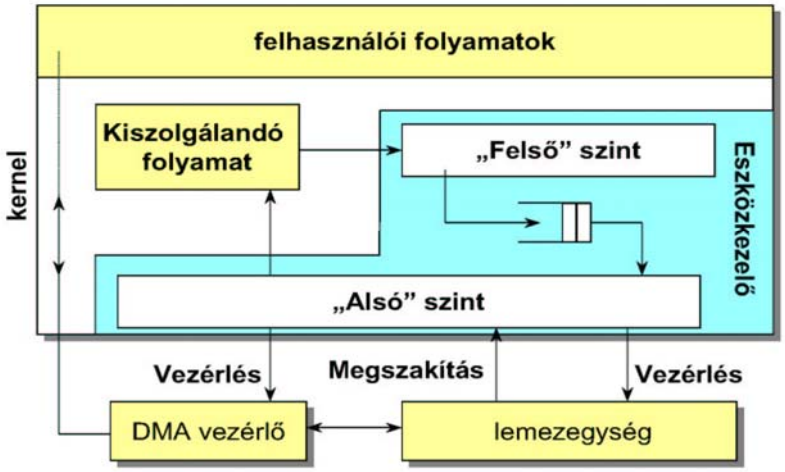
Az alapvető problémát az okozza, hogy a háttértárak sem azonosítási rendszerükben (CHS), sem tárolási rendszerükben (adatok szektor(csoport)okban) nem illeszkedik a felhasználói logikához (képzeljük el, ha az állományainkat 24 bites bináris számsorokkal kellene azonosítanunk, valamint minden állományról rendelkezni azzal az információval, hogy a háttértár mely részein helyeztük el…). Az operációs rendszer a fentiek kezelésére használja a fájl és a könyvtár fogalmát.

- **Állomány**: felhasználói szempontból összetartozó adatok megkülönböztető azonosítóval rendelkező csoportja a háttértáron (a felhasználói adattárolás egysége).
- **Könyvtár**: a felhasználói állományok csoportosítását lehetővé tevő logikai azonosítók (strukturált) rendszere a háttértáron (megvalósításában: adminisztratív állomány).
- **Kötet**: a háttértároló egyedi logikai azonosítóval rendelkező területe (az operációs rendszer számára a háttértár megkülönböztethető része).

Általában a háttértár teljes területe alkot egy kötetet, de az operációs rendszerek általában lehetőséget adnak ettől eltérő megfeleltetés kialakítására is: ugyanazon háttértár különböző részei (partícionálás révén) különböző kötetként is kezelhetőek, illetve különböző háttértárak részei „összevonhatók” a kezelés szempontjából egyetlen logikai egységgé. A kötet azonosítására egyes operációs rendszerek az angol ábécé betűit használják, mások megengedik a kötetek nevesítését.

Az operációs rendszernek az állományszervezéssel kapcsolatos feladatai: a felhasználói adatokat (kvázi) tetszőleges ideig változatlan formában tárolni képes tárolási rendszerben ezen információk célszerű elhelyezése, egyértelmű azonosíthatóságának biztosítása, visszakereshetőségének támogatása és hozzáférés-vezérlése.

Állománykezelő rendszer: az operációs rendszer állományszervezéssel kapcsolatos tevékenységeit megvalósító rendszerszolgáltatások csoportja. Az állománykezelő a rétegmodellben a felhasználói felület és az eszközvezérlők között helyezkedik el – azaz nem azonos és nem összetévesztendő a shell szinten megjelenő azonos nevű alkalmazásokkal (Commander programok, Intéző, stb.)!



2-11. ábra: Az OR állományszervezési komponensei

A felhasználó által megadott (egyedi) azonosító alapján az állománykezelő meghatározza a keresett információ fizikai helyét, és (szükség esetén az adott háttértár kezelésére szolgáló kiegészítő program – az **eszközvezérlő** - segítségével) utasítja a háttértár író/olvasó mechanizmusát a megfelelő művelet elvégzésére. Tekintve, hogy a számítógép és a háttértárak műveletvégzési sebessége jelentősen eltér, a gyakorlatban az állománykezelő és a hardver eszköz közé egy gyors átmeneti tárolót (**puffer**, „buffer", „cache") is beépítenek.

2.5.3.2. Állománykezelés

Azt, hogy az egyes operációs rendszerek mit tekintenek „egyedi azonosítónak", nem rögzítik szabványok, de általánosan elmondható, hogy a felhasználó valamilyen alfanumerikus karaktersorozattal hivatkozhat a tárolt adatokra – ez az állomány **neve**.

A névhasználat szabályai (amelyek szintén operációs rendszerenként változnak) határozzák meg, hogy az állományok neve milyen hosszú lehet (hány karakterből állhat), milyen szimbólumokat tartalmazhat (általában azok a karakterek, amelyekhez az operációs rendszer valamilyen egyéb tevékenységet rendel hozzá, tiltottak), különböznek-e a nevekben előforduló kis- és nagybetűk, adott esetben a név felépítésére is vonatkozhatnak megkötések.

Az operációs rendszertől függ az is, hogy milyen egyéb adatokat tart nyilván az egyes állományokról. A legáltalánosabban használt **tulajdonságok** a következők:

1. **típus**: mivel a tárolt információ elemzése nem az operációs rendszer feladata, de az egyes állományok tartalma alapvetően meghatározza a vele (rajta) elvégezhető műveletek körét, az operációs rendszerek általában az állomány nevét kiegészítik egy azonosítóval, ami segít annak meghatározásában, hogy milyen műveletek értelmezettek az adott adatcsoporttal. A DOS alapú operációs rendszerekben örökségeként szokás ezt a típusazonosítót **kiterjesztésnek** is nevezni.
2. **időbélyegek**: az állományok használatakor más és más időpillanatokban különféle tevékenységeket végezhetünk az adatokon, amelyek feljegyzése részben adminisztratív, részben biztonsági célokat szolgál. Általánosan használt időbélyeg az állomány **keletkezésének időpontja** (amikor az adatcsoport rögzítésre került a háttértárolón), de egyes operációs rendszerek rögzítik az állománnyal végzett műveletek (**utolsó megnyitás**, **utolsó módosítás**) időpontját is.

3. **méret**: az állományban tárolt információ mennyisége, általában bájtban kifejezve. Fontos tudni, hogy az állomány tartalma és a tárolásához szükséges hely nagysága általában nem azonos! (*Kérdés: miért?*)
4. **jellemzők** (attribútumok): operációs rendszerenként változó kiegészítő információk (általában 1 biten tárolt értékek), amelyek az operációs rendszer számára hordoznak járulékos információt az egyes állományokról – gyakorlatilag meghatározzák az operációs rendszer egyes szolgáltatásainak, műveleteinek hatását az adott állományra (pl. írásvédelem, röptömörítés alkalmazása, stb.).
5. **jogosultságok**: amennyiben az operációs rendszer támogatja, megadható (korlátozható) az egyes állományokon végezhető műveletek (pl. módosítás, törlés, megnyitás, tulajdonságok megváltoztatása, stb.) köre. Elsősorban a többfelhasználós rendszerekben alkalmazott tulajdonság, amely a felhasználók azonosításának képességével együttesen szabályozott állománykezelést tesz lehetővé.

A felhasználói munkavégzés során új állományokat hozunk létre, létező állományokkal végzünk műveleteket vagy megszüntetjük a feleslegessé vált állományokat. Az állományokkal elvégezhető műveletek alapvetően két csoportra oszthatók: az állomány tartalmával kapcsolatos és az állomány tulajdonságait meghatározó/megváltoztató műveletekre. (A későbbiekben látni fogjuk, hogy az utóbbi csoportba tartozó tevékenységek nem tekinthetők állományműveletnek.)

Állományműveletek:

1. **létrehozás**: az állománykezelő a háttértár alkalmas (méretű) területét lefoglalja és a háttértár foglaltságát tartalmazó táblázatban (ld. később) rögzíti az állomány azonosítására szolgáló adatokat.
2. **megnyitás**: ahhoz, hogy a háttértárolón elhelyezkedő adatokkal műveletet végezhessünk, az állományt előbb meg kell nyitni. A megnyitás során az operációs rendszer azonosítja az állományt (logikai név – fizikai elhelyezkedés összerendelése), ellenőrzi a kijelölt művelet végrehajtásához szükséges jogosultságok meglétét és egy **fájl leíró táblát** (*FCB – File Controll Block*) hoz létre, ami tartalmazza az állomány elhelyezkedésével és kezelésével kapcsolatos információkat – lényegében ezen a struktúrán keresztül kezelhető az állomány tartalma.

   A megnyitott állomány tartalma és az operatív memóriába közötti adatmozgatási művelet (irányultságtól függően) lehet **olvasás** (háttértár $\Rightarrow$ memória) vagy **írás** (háttértár $\Leftarrow$ memória). Az írási műveletek között különbséget kell tenni aszerint, hogy az állomány esetleg létező előző tartalma megváltozik az írási művelet során (**módosítás**) vagy a kiírásra kerülő információ az állomány korábbi végétől folytatólagosan helyezkedik el (**hozzáfűzés**). Azt, hogy az adott állománnyal milyen műveletet fogunk az adott feldolgozási folyamat során elvégezni, megnyitáskor kell megadni: ilyen értelemben beszélhetünk az állomány **megnyitási módjáról**. A megnyitási mód az adott feldolgozási folyamatban meghatározza az állománnyal végezhető műveletek körét (pl. egy olvasásra megnyitott állományba történő írási parancs végrehajtását az operációs rendszer vissza fogja utasítani).
3. **pozícionálás**: a következő írási vagy olvasási művelet elvégzésének helyének meghatározása. Az operációs rendszer egy mutató segítségével tartja nyilván,

hogy a feldolgozási folyamat az állomány tartalmának mely részénél jár (természetesen ez a mutató is az FCB-ben található), ennek a mutatónak a beállításával történik az állománynak a következő műveletben érintett részének a kijelölése.

4. **lezárás**: az FCB megszüntetése, az állomány tartalmának rögzítése a háttértáron. Lezárt állománnyal állományszintű művelet nem végezhető.

Az állományok kezelése során az állomány tartalmának feldolgozási módja szerint megkülönböztethetünk **szöveges** vagy **bináris módú** (az előbbinél a tárolt információ értelmezését a feldolgozó programnak kell elvégeznie, az utóbbinál az adatok feldolgozását az állományban tárolt vezérlőjelek – tipikusan ilyen a **„sor vége"-jel** – segítik), pozícionálás szempontjából pedig **soros** vagy **közvetlen elérésű** (az előbbinél a pozícionálás csak folyamatosan növekvő módon történhet (azaz az állomány egy adott bájtjának feldolgozásához az összes megelőző bájton végig kell haladnunk), az utóbbinál van lehetőség az állomány tetszőleges pontjának sorszám alapján történő közvetlen elérésére) állományokat.

Az állományok életciklusában (létrehozás – felhasználás: olvasás, módosítás, futtatás, stb. – megszűnés) a keletkezés lényegesen gyakoribb tevékenység, mint a megszűnés, így az egyes háttértárolókon az állományok száma általában nő – ez viszont felveti a tárolás és a visszakeresés néhány problémáját: pl. az egyértelmű azonosítás kényszere végett minden állománynak egyedi nevet kell adni, márpedig nagyszámú állomány esetén nem könnyű megjegyezni, mely neveket használtuk már fel; vagy hogy egy sok állományt tartalmazó listából a keresett állomány kiválasztása lényegesen bonyolultabb feladat. Célszerű tehát az állományok tárolását valamilyen rendszerbe foglalni.

2.5.3.3. Könyvtárkezelés

A háttértáron az adatok rendszerezetten helyezkednek el – ez azonban az operációs rendszer és a háttértár tárolási szerkezetének megfelelő rendszer, ami nem követi és nem tükrözi a az egyes állományok között a felhasználó számára jelentkező összefüggéseket. Az igény tehát az, hogy az adatok a tárolási szerkezettől független logika szerint legyenek visszakereshetőek: a felhasználó számára összetartozó információk a felhasználó számára (látszólag) azonos helyen legyenek elérhetők. A helyzet tehát ugyanaz, mint az állományoknál: ugyanaz a fogalom mást jelent és másként jelenik meg a számítógép és a felhasználó számára.

Könyvtár (mappa, katalógus, „*directory*"):

1. az operációs rendszer által használt adminisztratív célokat szolgáló állomány, amely a felhasználói információt tartalmazó állományok logikai csoportosítását a fizikai elhelyezkedésük tárolásával teszi lehetővé.
2. a (felhasználói) állomány elhelyezkedésének logikai struktúráját tükröző bejegyzés.

A könyvtárak tehát tulajdonképpen állományok (ami természetes is, hiszen az operációs rendszer számára minden, a háttértáron elhelyezkedő információ állomány), amelyek azzal a speciális tulajdonsággal rendelkeznek, hogy tartalmuk nem a felhasználó, hanem az operációs rendszer számára hordoz értelmezhető információt: azt, hogy azok az állományok, amelyek a felhasználó számára azonos helyen levőnek **látszanak**, hol helyezkednek el **ténylegesen** a háttértárolón.

A fentiek szerint tehát a könyvtár nem más, mint egy nyilvántartás – ebből viszont az következik, hogy valamilyen alkalmas tárolási szerkezetet kell kialakítani. Ahhoz, hogy a már ismert adatszerkezetek (sor, lista, tömb, verem, stb.) közül kiválaszthassuk a célunknak

leginkább megfelelőt, nézzük meg milyen elvárásaink lehetnek a nyilvántartásunkkal szemben?

- **Rugalmasság**: előre nem ismert, hogy a könyvtárban hány állomány információit fogjuk tárolni – sőt ez az érték a felhasználás közben folyamatosan változik.
- **Csoportosíthatóság**: tetszőleges szempontok szerint.
- **Gyors visszakeresés**: a könyvtárak szerepe abban áll, hogy segítsenek a nagyszámú állomány között megtalálni a számunkra fontosat.

A lehetséges tárolási szerkezetek közül az operációs rendszerek a **hierarchikus fa struktúrát** használják legelterjedtebben. Ez a struktúra egy körmentes, irányított gráffal írható le, amelynek pontosan egy kiinduló csúcspontja (**gyökér**) van és minden más pontjába pontosan egy út vezet a gyökértől. Az egyes csúcsokból tetszés szerinti számú (irányított) él vezet más csúcsokhoz, azt a csúcsot, ahonnan egy él kiindul, **szülőnek**, ahová vezet, **gyereknek** (**alkönyvtár**) nevezzük. A gyökérnek nincs szülője és minden gyereknek pontosan egy szülője van..

A könyvtárak névhasználatára általában az állományokkal azonos szabályok vonatkoznak (*kivéve a gyökérkönyvtárat: a gyökérkönyvtár nevét nem a felhasználó, hanem az operációs rendszer határozza meg*), típusuk és méretük általában nincs, az állományoktól egy attribútum különbözteti meg őket, a jogosultsági tulajdonságok (bizonyos értelemszerű megszorítások mellett) az állományokkal azonos módon értelmezhetőek a könyvtárakra is.

Hasonló a helyzet a könyvtárakon értelmezett műveletek vonatkozásában is (ne feledjük: az operációs rendszer számára a könyvtár is csak állomány!), azonban két lényegi különbséget mindenképpen ki kell emelnünk a könyvtárakon és állományokon elvégezhető műveleteket vizsgálva: a könyvtárakkal a felhasználó csak közvetett módon (az operációs rendszer szolgáltatásainak igénybe vételével, ún. **rendszerhívásokon** keresztül) végezhet műveletet, és ezen műveletek jelentős része állományokkal kapcsolatos művelet - amelyeket a fentiekben nem soroltunk fel állományokon „értelmezett” műveletekként. Ezért a könyvtárműveleteknél csupán a különbségek kiemelésére szorítkozunk.

Könyvtárakra vonatkozó műveletek:

1. **létrehozás**: lévén a könyvtár csak egy virtuális állomány, létrehozása nem jár tényleges helyfoglalással, csupán a szülőkönyvtár nyilvántartása bővül egy újabb bejegyzéssel.
2. **megnyitás**:
    a. az állományoknál tárgyalttal azonos értelmű művelet, a könyvtárbejegyzéseket tartalmazó fizikai állomány tartalmának hozzáféréséhez szükséges, a nyitott bejegyzés-állományon értelmezett műveletek és hatásuk:
        i. **olvasás**: a könyvtárban tárolt állományok (és/vagy könyvtárak) adatainak lekérdezése (listázása).
        ii. **írás**: a könyvtár egy adott állományára (könyvtárára) vonatkozó leíró részek módosítása. Ide tartozik:
            1. állomány **létrehozása** (az állomány létrejöttekor készül róla egy feljegyzés a könyvtár nyilvántartásában);

2. állományok **tulajdonságainak** (név, méret, stb.) **megváltoztatása** (mivel ezek a információk nem az állományban tárolódnak);
3. állományok **törlése** (a tartalmazó könyvtár állományra vonatkozó bejegyzése módosul, nem történik meg az állomány tényleges eltávolítása a háttértárról)
4. állomány helyének megváltoztatása (**másolás**: az állomány tartalma ismételten található meg a háttértár két különböző helyén; **áthelyezés**: az állomány logikai helyének megváltoztatása – általában nem jár az állomány fizikai helyének megváltozásával, csupán az érintett könyvtárak adminisztrációs bejegyzéseinek módosulásával)

b. logikai (felhasználói) értelemben ugyanígy nevezzük egy könyvtár kiválasztását, aktuálissá tételét is (szokás **könyvtárváltásnak** is nevezni). Hatására azok az állományok, amelyek ebben a könyvtárban találhatóak, hozzáférhetőek: megjeleníthetők (listázhatók), feldolgozhatók. Ez a művelet helyettesíthető az *elérési út* (ld. később) alkalmazásával.

(*Kérdés: a fenti műveletekben az állomány és a könyvtárfogalmat analóg módon használtuk, hiszen a művelet ugyanazt jelenti (al)könyvtárra vagy állományra vonatkoztatva – kivéve a törlés műveletét: a könyvtár törlése általában feltétel(ek)hez kötött művelet, a leggyakoribb ilyen feltétel az, hogy csak olyan könyvtár törölhető, amely nem szerepel az aktuális könyvtártól a gyökérkönyvtárig vezető útvonalban. Miért?*)

2.5.3.4. Hivatkozások

Elérési út: egy könyvtár helyének megadása a hierarchikus könyvtárstruktúra szerkezetének megfelelően.

Az elérési út a könyvtár azonosítására szolgál. Mivel a gyökértől minden könyvtárhoz pontosan egy út vezet, ennek az útvonalnak a megadása egyértelműen azonosítja a könyvtárat. A megadás során az elérési út könyvtárait az operációs rendszer speciális szimbóluma (általában a / vagy a \) választja el egymástól.

Abszolút elérési út: a gyökérkönyvtártól induló és a hivatkozott könyvtárig az összes tartalmazó könyvtárat a hierarchiában elfoglalt helyének megfelelő sorrendben megadó útvonal-leírás.

Az abszolút elérési út használatával minden könyvtár (és minden állomány) egyértelműen azonosítható, azonban ez a fajta megadási mód egy összetettebb (már akár 4-6 szintű) hierarchia esetén meglehetősen kényelmetlen lehet. Egyszerűbbé tehető a hivatkozás, ha a könyvtár helyét nem a gyökérhez, hanem a pillanatnyilag használt könyvtárhoz képest adjuk meg (*ld. lokalitás elve: ha egy művelet egy adatra hivatkozik, akkor nagy valószínűséggel (és hamarosan) a hivatkozott adat környezetében található adatokra is szüksége lesz*). Ez viszont egyrészt azt jelenti, hogy nyilván kell tartanunk az aktuálisan használt könyvtárat (szokás **munkakönyvtárnak** is nevezni), másrészt szükségünk lesz olyan hivatkozások bevezetésére, amelyek a „visszafelé” hivatkoznak: a szülőkönyvtárakra.

Az elterjedt hivatkozási könyvtár-szimbólumok a ’**.**’ („1 pont”) a könyvtár saját adataira, a ’**..**’ („2 pont”) pedig a szülőkönyvtárra mutató speciális hivatkozás.

Relatív elérési út: a hivatkozott könyvtár elérését megadó leírás, amely az aktuálisan használt (munka-) könyvtártól indul.

2.5.3.5. Állományszervezési stratégiák

Az előzőekben megismert logikai adattárolási fogalmak nyilvánvalóan nem felelnek meg a háttértár tárolási szerkezetének. Láttuk, hogy az információ állományokban, az állományok pedig a fizikai elhelyezkedéstől független könyvtárakban tárolódnak. Az egyes állományok fizikai elhelyezkedésével kapcsolatos információt az állományt tartalmazó könyvtár megfelelő bejegyzése tartalmazza.

Folytonos állomány-elhelyezési módszerek

Az állományok tárolása során az operációs rendszer elsődleges feladata, hogy a háttértáron az adatok méretének megfelelő nagyságú tárolóterületet találjon. Mivel az állományok mérete általában meghaladja a háttértár blokk-méretét, ezért nem is blokk, hanem blokkok meghatározásáról van szó. A későbbi feldolgozás és adminisztráció szempontjából szerencsés, ha a tárolóterület a háttértároló egy összefüggő (folyamatos címekkel rendelkező) blokk-csoportja, hiszen ilyenkor olvasható leggyorsabban vissza az állomány tartalma, és a nyilvántartásban mindössze a kezdetét kell feljegyezni. (Más kérdés, hogy az állomány bővítése már alkalmasint problémákat vet fel (mi legyen, ha a következő blokk már egy másik állomány által használt?) és a törlések miatt a rendelkezésre álló hely szétdarabolódóik (**töredezettség**), és előfordulhat, hogy bár összességében még lenne elég hely, mégsem tudunk egy újabb állományt kiírni).

Kérdés csak az, hogy hogyan határozzuk meg, hogy hová kerüljön az állomány? Mivel ilyen információval nem rendelkezik, az operációs rendszer végigolvassa a háttértárat és minden blokk esetében megvizsgálja, hogy szabad vagy foglalt –gondoljunk csak bele, ha ez ténylegesen a lemez végigolvasását jelentené, mennyi időt venne igénybe? Ehelyett indulásakor az operációs rendszer készít egy táblázatot (amit a memóriában tárol és kezel), aminek minden egyes cellája a háttértár egy-egy blokkjának felel meg és tartalma (mondjuk) 0 vagy 1 aszerint, hogy az adott blokk felhasználható még vagy már foglalt. Ekkor nyilvánvalóan elegendő csupán ebben a táblázatban megkeresni a megfelelő méretű szabad helyet – ez esetben természetesen az állományokon végzett műveleteknek (létrehozás, törlés) megfelelően folyamatosan aktualizálni kell ezt a leírótáblát is.

További problémát okoz, ha a háttértáron több alkalmas hely is található. Ez esetben az operációs rendszer valamilyen szempont szerint választ a rendelkezésre álló helyek között, ezt nevezzük az operációs rendszer **állomány-elhelyezési stratégiájának**. A legjellemzőbb stratégiák:

- **legelső megfelelő** (*First Fit*): az első alkalmas méretű tartományt foglalja le. Leggyorsabb, de nem feltétlen hatékony.
- **legjobban illeszkedő** (*Best Fit*): azt a tartományt használja fel, amely a legkisebb mértékben tér el az állomány méretétől. Lassú, de hatékony: a legkevesebb veszteséget eredményezi.
- **legrosszabbul illeszkedő** (*Worst Fit*): az előző ellentéte, a legnagyobb eltérésű területen helyezi el az állományt – így biztosítja, hogy továbbra is (relatíve) nagy blokk-csoportok maradjanak felhasználhatóak.

A folytonos állományszervezési módszerrel kapcsolatban alapvetően két problémát kell megemlítenünk: az egyik a már említett töredezettség (az állományok törlésekor a kialakuló szabad területek – bár együttes méretük esetleg elég nagy lehet – mivel nem folytonosan helyezkednek el, nem használhatók újabb állományok elhelyezésére), a másik a

kötött méret: a foglalás során kialakított állományméret nem növelhető (azaz egyrészt létrehozáskor meg kell adni az állomány végleges méretét, és a későbbiekben az állomány nem bővíthető).

Nem folytonos állomány-elhelyezési módszerek

Lényegesen jobb helykihasználás érhető el, ha az állományokkal kapcsolatban nem követeljük meg a folytonos elhelyezkedést – ez esetben egyrészt bonyolódik az adminisztráció (nem elég azt nyilvántartani, melyik blokkban kezdődik az állomány, hanem azt is tudni kell, hogy melyikben folytatódik!), másrészt lassul az adatok visszaolvasása (az állomány részei a háttértár különböző helyein lehetnek), viszont gyakorlatilag a teljes kapacitás kihasználható, és az állományok tetszés szerint bővíthetőek.

Aszerint, hogy az állományok egymást követő részeinek nyilvántartására milyen módszert alkalmazunk, két stratégiáról beszélhetünk:

- **Láncolt lista**: a szabad tartományok meghatározásánál használt elv alapján a lemez blokkjainak megfelelő elemszámú táblázatot hozunk létre, de most az egyes mezőkben azt tároljuk, hogy amennyiben a háttértáron az adott sorszámú blokkban egy állományhoz tartozó adatcsoport van, akkor melyik sorszámú blokk tartalmazza ugyanazon állomány következő adatcsoportját. Ezt a táblázatot nevezzük **fájl elhelyezkedési táblának** (*File Allocation Table, FAT*).

Az állomány első részét tartalmazó blokk sorszámát a könyvtárbejegyzésbe írjuk, az állomány utolsó blokkjának a jelölésére speciális sorszámot (0, -1) használunk.

A következő ábra a láncolt elhelyezés stratégiát szemlélteti:

Fájl: valami.txt

Könyvtárbejegyzés

név	méret	dátum	jellemzők	#1blokk
valami.txt	22 845	2002-01-12	10110001	74

FAT

59	60	61	...	74	...	88	89	...	93	94	95
66	00	94		61		60	77		89	95	88

Háttértár

59	60	61	...	74	...	88	89	...	93	94	95
	#6	#2		#1		#5				#3	#4

2-12. ábra: FAT alapú állományszervezési módszer működési elve

A módszer előnye, hogy minden blokk kihasználható, nincs szükség a helykereső stratégiák alkalmazására (az első szabad blokktól el lehet kezdeni az állomány elhelyezését),

valamint, hogy a láncolt adatstruktúrában a bővítés és a törlés műveletét egyszerű elvégezni (csak az elhelyezkedési lista megfelelő hivatkozásait kell módosítani).

Hátránya a FAT szerepének kizárólagossága (ha megsérül, a háttértáron tárolt információk visszanyerése elég reménytelen feladat – ennek megelőzésére általában két példányban tárolódik), valamint hogy egy akár csak egyetlen állomány feldolgozásához is az egész táblázatot a memóriában kell tárolni (gyakorlatban az állományműveletek gyakorisága miatt a FAT folyamatosan a memóriában helyezkedik el), és az, hogy nem támogatja az állomány tartalmának közvetlen elérését (a x. adatcsoport helyének meghatározásához végig kell olvasni a tárolási listát).

- **Indextábla**: ebben az esetben minden állomány saját fájl elhelyezkedési táblával rendelkezik. A táblázat egyes mezői sorrendben tartalmazzák az adatblokkok fizikai címét (az elhelyezkedést leíró táblázat neve az **indextábla**), a táblázat helyét pedig az állományhoz tartozó könyvtárbejegyzésben adjuk meg.

A következő ábra az indextábla alapú állomány-elhelyezési stratégiát mutatja be:

fájl: valami.txt

könyvtárbejegyzés

név	méret	dátum	jellemző	indextábla
valami.txt	22 845	2002-01-12	110011101	93

indextábla

74	61	94	95	88	60			

háttértár

59	60	61	...	74	...	88	89	...	93	94	95
	#6	#2		#1		#5			I	#3	#4

2-13. ábra: Az indextábla alapú állományszervezési módszer működési elve

A módszer előnye, hogy az indextábla esetleges sérülése csak az adott állomány elvesztését eredményezi, a mérete jelentősen kisebb a FAT-nál, és hogy a lehetőséget ad az állomány tartalmának közvetlen elérésére (az állomány x. adatcsoportja az indextábla x. mezőjében szereplő című blokkban található).

A problémát jelent viszont, hogy az állományok maximális méretét meghatározza az indextábla mezőinek a száma (ennek kivédésére az indextáblákat láncolása szolgálhat: az indextábla utolsó mezője nem az adatállomány utolsó blokkjára, hanem a következő indextábla címére mutat – ily módon több indextábla összefűzhető), továbbá hogy míg a FAT egyben a háttértár foglaltságának nyilvántartására is alkalmas, ennék a módszernél külön kell gondoskodni a szabad és foglalt blokkok megkülönböztetéséről, és hogy az állomány tartalmának módosulása (bővülés vagy egyes közbenső adatcsoportok törlése) az indextábla újra-összeállítását igényli.

Összességében látható, hogy a két dinamikus stratégia lényegében egymás tükörképe: eltérésük abból fakad, hogy az egyik megvalósításban bizonyos tevékenységeket emeltek ki, mint célokat, és ezeknek rendeltek alá más feladatokat – a másik esetben pedig pont fordítva.

Ellenőrző kérdések

1. Melyek az operációs rendszerek legfontosabb alapfeladatai (szolgáltatásai)?
2. Hogyan definiálná az operációs rendszer fogalmát?
3. Miben áll a rétegmodell jelentősége az operációs rendszerek szerkezetének vizsgálata során?
4. Melyek a legjellemzőbb rétegszolgáltatások?
5. Hogyan működik a megszakítási rendszer?
6. Mi a kernel?
7. Melyek az operációs rendszerek legfontosabb kernel-szintű szolgáltatásai?
8. Milyen folyamat-állapotokat ismer?
9. Ismertesse a folyamatok életciklusát!
10. Melyek a karakteres felhasználói felület általános jellemzői?
11. Milyen részekből épül fel egy utasítás?
12. Milyen eszközöket, módszereket ismer karakteres felületen több utasítás együttes végrehajtására vonatkozóan?
13. Melyek a grafikus felhasználói felület alapelemei?
14. Mutassa be a grafikus felület eseményvezérelt működési modelljét!
15. Mire szolgál és hogyan épül fel az X-Window architektúra?
16. Milyen általános utasítás-szerkezeteket ismer?
17. Hasonlítsa össze a fordító és az értelmező módú parancsértelmezők működését!
18. Hogyan definiálná az operációs rendszerek „virtuális gép” koncepcióját?
19. Milyen tényezők határozzák meg a háttértáron tárolt adatok elérését? Hogyan lehet ezeket befolyásolni?
20. Magyarázza meg az állomány, könyvtár, kötet fogalmakat!
21. Mit értünk az állományszervezés logikai és fizikai szintje alatt? Hogyan kapcsolható össze ez a két tevékenységi kör?
22. Melyek az állományszervezési rendszer kialakítása során szem előtt tartandó legfontosabb szempontok?
23. Hogyan működnek a folytonos állományszervezési módszerek?
24. Ismertesse a láncolt lista (FAT) állományszervezést megvalósító háttértárak működését!
25. Ismertesse az indextábla alapú állományszervezést megvalósító háttértárak működését!
26. Hasonlítsa össze a láncolt lista és az indextábla alapú állományszervezési módszereket: adjon példákat előnyös és hátrányos tulajdonságaikra!

3. OPERÁCIÓS RENDSZEREK A GYAKORLATBAN

Bevezetés

Az eddigiekben megismerkedtünk az operációs rendszerek általános működésének legfontosabb alapelveivel, a leggyakrabban alkalmazott módszerekkel. Természetesen ezek az ismeretek is fontosak, de felhasználói szempontból lényegesen nagyobb jelentőséggel bírnak az egyes rendszerek használatához szükséges gyakorlati ismeretek. A következőkben a számítógépes munkavégzés során legelterjedtebben alkalmazott operációs rendszerek kezelésével kapcsolatos gyakorlati ismereteket mutatjuk be. A rendszerek kiválasztásakor az elterjedtség mellett az adott rendszerben alkalmazott alapelvek sokszínűsége, a gyakorlati megvalósításokban előforduló hasonlóságok és különbségek bemutathatósága is szempont volt. Nem gondoljuk azonban, hogy ezeket az ismereteket pusztán a jegyzet áttanulmányozásával el lehet sajátítani! Arra buzdítjuk az olvasót, hogy a megismert módszereket, alkalmazásokat, technikákat lehetőség szerint azonnal próbálja ki, gyakorolja be, mert csak így beszélhetünk készségszintű számítógép-kezelésről – márpedig az operációs rendszerek gyakorlati alkalmazása ezt kell, hogy jelentse!

A kiválasztott operációs rendszerek megismerése a bemutatott példákon keresztül azért sem lesz (lehet) egyszerű feladat, mert mindkét operációs rendszer számos eltérő változatban érhető el. A Linux alapú rendszerek bemutatásakor részben azokra az általános elvekre és eszközökre próbáltuk meg a hangsúlyt fektetni, amely több disztribúcióban is hasonló módon megtalálható. A Windows alapú rendszerek bemutatásakor pedig a jegyzet írásakor aktuális változat (a Windows XP) sajátosságai közül megpróbáltuk kiemelni azokat, amelyek más változatokban esetlegesen máshogy jelenhetnek meg.

Ezzel együtt is előfordulhatnak olyan képernyőképek, utasítások, amely az Ön által használt rendszerben nem pontosan a leírtaknak megfelelően jelenik meg vagy működik – a cél ez esetben nem a mechanikus másolás, hanem a működésnek, a mögöttes tevékenységnek a megértése kell, hogy legyen. Ehhez pedig kitartásra és sok gyakorlásra lesz szüksége. Kívánjuk, hogy mindkettőből rendelkezzen elegendővel!

3.1. UNIX/Linux alapú rendszerek

3.1.1. Történelem

3.1.1.1. UNIX

Bevezetés

Az 1960-as évek végén komoly erőfeszítéseket tettek, hogy új operációs rendszertechnikákat fejlesszenek ki. 1968-ban a General Electric, az AT&T Bell Laboratories és a Massachusetts Institute of Technology kutatóinak egy csoportja a MULTICS elnevezésű operációs rendszer témájú kutatási programba kezdett. A MULTICS számos új elemet tartalmazott a fájl kezelés, a többfeladatos (multitasking) feldolgozás és a felhasználói felület területén.

A UNIX operációs rendszer első változatát (amely a MULTICS számos előnyét felhasználta) 1969-ben készítette el Ken Thompson az AT&T intézetnél egy leselejtezett PDP-7 számítógépen. A UNIX operációs rendszert eredetileg kutatók számára fejlesztették ki. Egyik fő célkitűzése az volt, hogy az operációs rendszer képes legyen a kutatók változó igényeit támogatni.

Miután az AT&T más munkatársai is jó lehetőségeket láttak a programban, az AT&T szoftverfejlesztőinek egy része elkezdett ezzel komolyabban foglalkozni, és egyre újabb, fejlettebb változatokat hoztak ki. 1970-ben Dennis Ritchie és Ken Thompson újraírták a UNIX programkódját, C nyelven (a C programnyelvet is Dennis Ritchie fejlesztette ki). A C programnyelv lehetővé tette, hogy a UNIX egyetlen verzióját kelljen csak elkészíteni, amelyet a C fordítóval lehetett a különböző számítógép típusokra lefordítani. UNIX hordozhatóvá vált azaz telepíteni lehetett különféle számítógépen anélkül, hogy a forráskódját jelentősen meg kellett volna változtatni.

Mivel a UNIX rendszert C nyelven készítették, nagyon könnyű volt átírni egy új hardverre, ezért nagyon hamar elterjedt az egész világon. Elterjedését segítette az is, hogy az első pár évben a rendszer teljes forráslistája bárki számára (ingyen) hozzáférhető volt. Több híres egyetem számítástudománnyal foglalkozó tanszéke is ingyen jutott a teljes operációs és fejlesztői rendszerhez. Ennek egyik következménye az lett, hogy a legtöbb helyen az egyetemi oktatásban is ezen rendszert kezdték használni. A UNIX hetedik változatának megjelenése után az AT&T felismerte a UNIX piaci lehetőségét, és a forráskód már csak a magas jogdíjak megfizetése ellenében lett hozzáférhető.

Kezdetben az AT&T UNIX változata (1972-ben jelent meg az első hivatalos változat) mellett jelentős volt a Berkeley Egyetem számítógép tudományokkal foglalkozó tanszékének a BSD UNIX (Berkeley Software Distribution) változata, amely 1975-ben jelent meg először. 1980-ban a Microsoft kifejlesztette a UNIX PC-s változatát, a XENIX-et. 1982-ben a AT&T kiadta első kereskedelmi változatát a UNIX-nak, a System 3-at, melyet a System V követett. A System V egy olyan kereskedelmi termékké vált, amely már komoly támogatással bírt. Eközben a UNIX BSD változata is számos fejlesztésen esett át. Az 1970-es évek végén a DARPA (Department of Defense's Advanced Research Project Agency) egyik kutatási programjának alapjául választották a BSD UNIX-ot. 1983-ban megjelent a BSD 4.2, amely már számos eredményét tartalmazta a DARPA-nak. Ilyen volt a bonyolult fájl kezelés és az Internet alapjául is szolgáló TCP/IP protokollra épülő hálózatkezelés. A BSD 4.2 verziót adaptálta a Sun Microsystems is, amely a későbbiekben a SUN UNIX-os operációs rendszerének a Sun SOLARIS-nak képezte a vázát.

A UNIX különböző verzióinak megjelenése szükségessé tette az egységesítést. Az 1980-as évek közepén két standard maradt: az egyik az AT&T Unix, a másik a BSD UNIX. Az

AT&T a UNIX-ot egy új szervezetre, a Unix System Laboratories-ra bízta, amely egy standard rendszer érdekében integrálta a Unix jelentősebb verzióit. A Unix System Laboratories 1991-ben kibocsátotta a System V4-et, amely egyesítette a System V3, a BSD 4.3, a Sun OS és a Xenix főbb jellemzőit. Válaszul több más társaság (IBM, SGI, Hewlett-Packard, Digital, stb.) megalakította az OSF-et (Open Software Foundation), hogy létrehozzak a saját standard Unix verziójukat. Így két standard kereskedelmi verziója jött létre a Unixnak: az OSF és a System V4. 1993-ban az AT&T eladta Unix érdekeltségét a Novell-nek, így a Unix System Laboratories a Novell UNIX System's Group része lett. A Novell kiadta a saját Unix verzióját, a UnixWare-t, amely a System V4-en alapul. (1995-ben a Novell is megvált a Unix érdekeltségétől, így az AT&T standard jelenleg az SCO tulajdonában van.)

A különféle fejlesztések ellenére a UNIX nagy és komoly hardware támogatást igénylő operációs rendszer maradt. A UNIX számos verzióját munkaállomásokra tervezték. A Sun OS-t Sun munkaállomásokra, az AIX-t IBM munkaállomásokra, az IRIX-et SGI munkaállomásokra. A személyi számítógépek teljesítményének növekedésével a UNIX PC-s fejlesztése is elkezdődött. A XENIX és a System V/386 a UNIX IBM kompatibilis személyi számítógépeire készült. Kezdetben az AUX, a UNIX Macintoshon futó változata volt, napjainkban MAC OS X névvel egyesíti az Apple korábbi grafikus operációs rendszerének és a UNIX-nak az előnyeit.

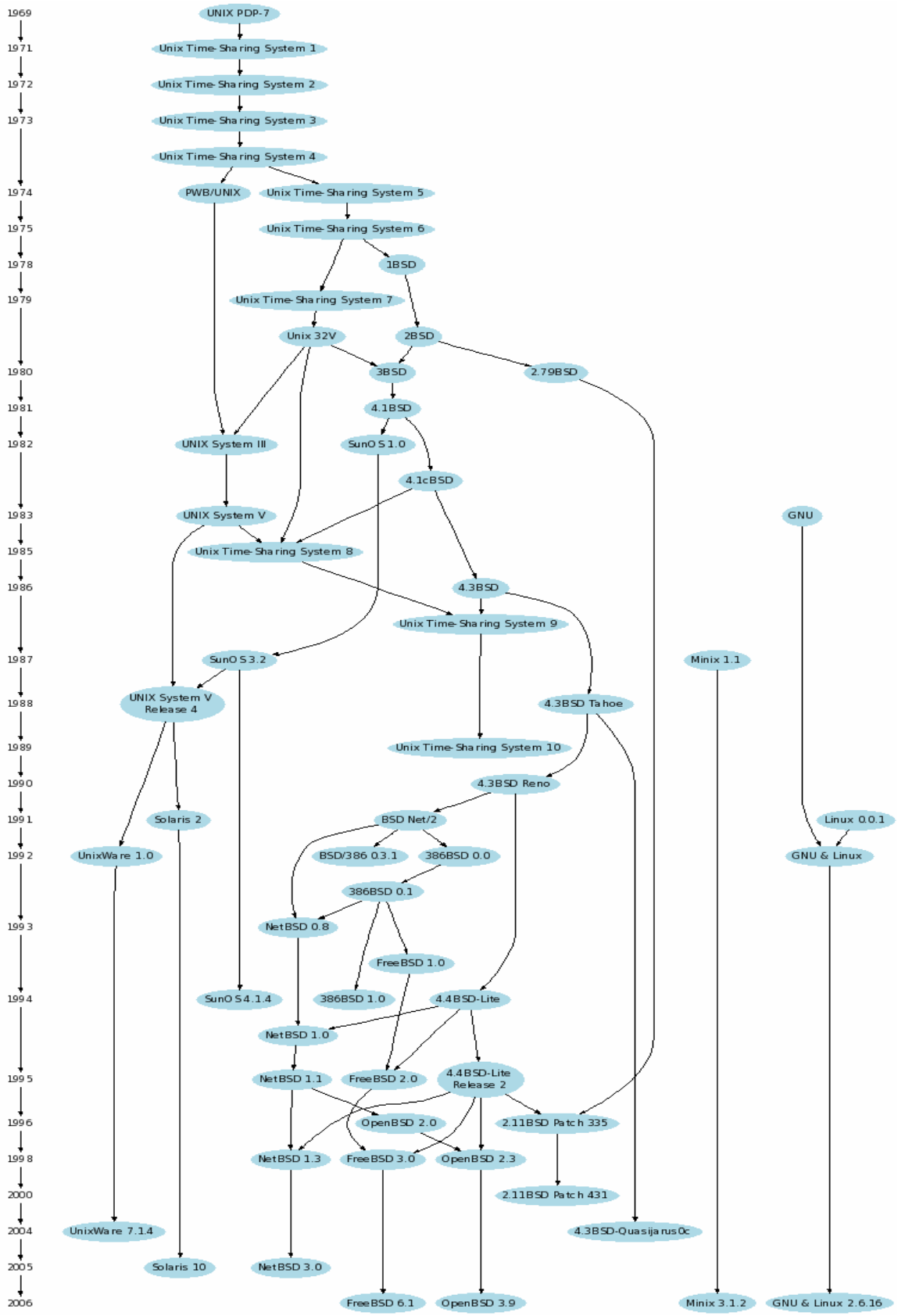
A Linux-ot Intel alapú személyi számítógépekre tervezték, amely kifejlesztése a Helsinki Egyetem egy hallgatójának, Linus Torvaldnak a személyes projektjeként indult. A Linux elődje a egy Minix nevű program volt, amely jól szemléltette a különféle UNIX jellemzőket. A Minixet Andrew Tannebaum professzor készítette, és az Interneten keresztül széles körben terjesztette. Linux egyik célja volt, hogy a Minix-használók számára a UNIX egy hatékony PC-s verzióját hozza létre. 1991-ben bocsátották ki a 0.11-es verziót.

Az IEEE (Institute of Electrical and Electronics Engineers) kifejlesztett egy független UNIX szabványt az ANSI (American National Standards Institute) számára, amelyet POSIX-nek (Portable Operating System Interface for Computer Environments) neveznek. A szabvány definiálja, hogy egy UNIX rendszernek hogyan kell működnie.

Jelenleg is elég sokféle UNIX van forgalomban (*3-1. ábra.*). Általában a nagyobb számítógépgyártó cégeknek van saját UNIX-a (*3-1. táblázat*):

3-1. Táblázat: jelentősebb UNIX rendszerek

Forgalmazó (fejlesztő)	UNIX neve
IBM	**AIX**
Hewlett-Packard	**HP-UX**
Silicon Graphics	**Irix**
Sun Microsystems	**Solaris, SunOS**
Novell	**Unixware, SuSE**
Apple	**MacOSX**



3-1. ábra: a UNIX rendszerek „családfája”

3.1.1.2. Linux

A Linux egy 32/64 bites, POSIX szabványt követő UNIX változat, amely eredetileg csak IBM PC gépeken futott (80386 vagy jobb processzor esetén), de mára nagyon sok hardverre adaptálták. A különböző hardverekre a rendszer kidolgozottsága eltérő fokú, de mindegyik esetén legalábbis összemérhető hatékonyságú és megbízhatóságú az azon a gépen szokásos operációs rendszerekkel. (Egyes vélemények szerint sok szempontból jobb is: pl. x86 alapú PC-k esetében sebesség és megbízhatóság tekintetében túlszárnyalja az elterjedtebb rendszereket.) A 32 bites változatok mellett léteznek 64 bites (Sun) és 8 bites (I80xx processzorokra) Linux verziók is – bár érdekes, hogy míg az Ultra Sparc processzoron futó 64 bites Linux teljesen stabilnak tűnik, addig a Sun még nem tudott előállni stabil 64 bites operációs rendszerrel ugyanerre a processzorra.

A Linux egy valódi 32/64 bites többfelhasználós (multiuser), többfeladatos (multitasking) operációs rendszer. A gondos programozás miatt ritka, hogy két program (pontosabban: két „process", folyamat – ld. később) zavarja egymást, így kitűnően alkalmas programfejlesztésre is. A Linux rendelkezik a szokásos funkciókkal: virtuális memória, merevlemez gyorsítótár, memórialemez, Internet hozzáférés, a leggyakoribb hardverelemek (CD-olvasó/író, nyomtató, IDE és SCSI lemezek, stb.) kezelése.

A rendszer kidolgozottsága olyan fokú, hogy egyre több helyen alkalmazzák UNIX munkaállomásként, vagy hálózati szerverként. Mindkét esetben hatalmas előny lehet más programokkal szemben a nagyfokú megbízhatóság és az alacsony ár, valamint a hordozhatóság: mivel nagy a hasonlóság a Linux és a nagyszámítógépek operációs rendszerei közt, egy IBM PC-n (Linux alatt) futó program könnyen átvihető pl. egy Sun SPARC gépre, de gondos programozás esetén akár egy CRAY szupergépre is.

Joggal merülhet fel a kérdés: ha mindez így van, miért nem Linuxot használ mindenki a világon. Mivel a Linux legtöbb változata szabad-terjesztésű, így a programozók általában nem vállalnak felelősséget azért, hogy az általuk írt rész működni fog. Ez sokakat visszariaszt – az üzleti életben legalábbis mindenképpen –, így ki sem próbálják a rendszert. Hasonlóképpen megfigyelhető, hogy a Linux mögött általában nincs egy „nagy" cég: nincs biztosíték arra, hogy a rendszer fejlesztése nem marad abba, nem megfelelő a marketingje, stb.

Az, hogy ezek ellenére a Linux-felhasználók száma milliókban mérhető, azt jelzi, hogy érdemes erre a rendszerre odafigyelni, és a számítástechnika történetének érdekes, és ma is élő színfoltját jelenti ez a rendszer, és az a mozgalom, ami körülötte kialakult.

A Linux története

A Linux fejlesztésének kezdetén Linus Torvalds a 80386 processzor védett módú (protected mode), feladat-váltó (task-switching) lehetőségeivel szeretett volna megismerkedni. A program fejlesztése egy korábbi PC-s UNIX, a Minix alatt történt, eleinte assembly-ben. A fejlesztés során felmerülő problémák előbb csak megvitatására, később megoldására Linus elhatározta, hogy az Interneten keresztül bevonja a fejlesztésbe a szabad kapacitással rendelkező programozókat.

A 0.12-es változat 1992. január 15-én látott napvilágot, néhány bővítéssel: Már volt init/login szolgáltatás (nem root-ként kellett először bejelentkezni, és inicializálni a rendszert), közeledett a POSIX szabványhoz, virtuális memóriát is használt és kissebb korrekciókat tartalmazott. Mivel ez egy elég stabil változat lett, elkezdődött a Linux hódítása. Szintén ehhez a változathoz kapcsolódik a Linux fejlesztésének kiszélesedése: a 0.12-es már lényeges részeket tartalmazott, melyeket nem Linus Torvalds írt.

A 0.95-0.99 rendszermagra épülő rendszereknek óriási népszerűségük volt. Bár a 0.95-ös verziótól kezdve a szolgáltatások száma, a megbízhatóság, és sok egyéb szempontból jelentős javulás következett be, és hihetetlenül sokan használták ezeket a rendszermagokat, az 1.0 verziószámot csak akkor merték kiadni, amikor a POSIX szabvánnyal való kompatibilitás kielégítővé vált.

A POSIX szabványosítás megfelelő szintű elérésével 1994. márciusában megjelent az 1.0.0 sorszámú kernel. Ekkortól kezdve egy speciális sorszámozási eljárást vezettek be a fejlesztők: a verziószámot három, ponttal elválasztott nem-negatív egész jelzi. Az első a fő verziószám, ami csak a rendszermag lényegét érintő változásoknál vált eggyel nagyobbra. A második szám elég speciális jelentésű: ha páros, akkor stabil, tesztelt kernelről van szó, amit bárkinek ajánlanak használatra, míg a páratlan szám tesztváltozatot jelöl, amit inkább azoknak javasolnak, akik tesztelni, fejleszteni szeretnék a kernelt (akiknek nem számít, ha a rendszer néha „elszáll"). A harmadik szám a kisebb módosításokat jelzi.

Ez a verziószámozási rendszer a 2.6-os kernel (2003) megjelenéséig volt érvényben, amikortól is nincs lényeges különbség a stabil és a tesztverzió között. Szintén a 2.6-os verziótól vezette be a Linux a négyjegyű verziószámok rendszerét, a negyedik jegy a kernelben felfedezett egyes hibák javításait tartalmazó változásokat jelöli (amelyek jelentőségükben nem érik el a „kisebb módosítás" minősítést). Ennek megfelelően a kernel aktuális verziószáma 2.6.16.11.

1996. augusztusában jelent meg a 2.0.0 sorszámú rendszermag. Ennek fő újítása a modulok megjelenése volt: a kernel bizonyos részei modulként is elkészíthetők, és ezek a modulok akár automatikusan, akár kézzel betölthetők a memóriába, ahonnét a rendszer kiveszi őket, ha régóta nem használjuk. Ilyenek lehetnek pl. a nyomtató- vagy floppi-vezérlők, a nem-Linux fájlrendszereket kezelő részek, mert ekkor ezek csak addig foglalják a memóriát, amíg éppen használjuk őket.

Ezzel az az érdekes helyzet állt elő, hogy a rendszermag memóriaigénye kisebb lett, míg hatékonysága és megbízhatósága megnőtt!

A Linux-terjesztések (disztribúciók)

A Linuxot kezdetben pusztán az Internetről lehetett beszerezni, és az installálás nem volt túl könnyű. Ekkor a rendszert még csak a számítógéphez nagyon értők használták. A népszerűség növekedésével azonban igény mutatkozott olyan programcsomag-rendszerre, amely a kevésbé szakértő számára is lehetővé teszi a telepítést. Ez volt az oka a Linux-terjesztések (disztribúciók) megjelenésének.

Általános tendenciaként azt említhetjük, hogy az egyre későbbi disztribúciók egyre jobban megkönnyítik a felhasználó dolgát.

Igaz, ennek ára van: egyrészt a legautomatikusabb változatok pénzbe kerülnek, másrészt egy automatikus telepítés sohasem olyan gazdaságos, mint egy kézi vezérlésű. Tehát egy automatikus telepítéskor felkerülhetnek felesleges programok is, vagy a konfiguráció nem a legjobban illeszkedik a rendszerhez, viszont a telepítés elkezdése és a rendszer használatba vétele közt sokkal kevesebb idő telik el, és az új programváltozatok is könnyen telepíthetők lesznek.

Történetileg az első, világméretekben elterjedt disztribúció a „Slackware" volt. Ez megkönnyítette a rendszer telepítését, így nemcsak számítógép-specialisták tudták feltenni a Linuxot a gépükre. Ez azonban nem azt jelenti, hogy a telepítés könnyű lenne: elég sok dokumentációt kell elolvasni annak, aki Slackware-t akart telepíteni.

Manapság CD-ről történik a legtöbb installáció, de sokszor az Interneten keresztül, valamilyen közeli szerverről történik a telepítés.

Manapság elsősorban a RedHat és a SuSE disztribúciók tekinthetők elterjedtnek – azzal a kiegészítéssel, hogy mindkét disztribúcióból számos újabb (ezek alapjait átvevő, azokat továbbfejlesztő vagy kiegészítő: Debian, Caldera, stb.) verzió jelent meg és létezik. Ezek a disztribúciók a könnyű telepíthetőséggel, a fejlett (és igény szerint testre szabható) csomagkezeléssel nyújtanak többet a hagyományos Linux verziókhoz képest.

Külön említést érdemelnek a „load-and-run” (telepítés nélkül is működő) típusú disztribúciók: az operációs rendszer méretcsökkenése lehetővé tette az egyetlen optikai adathordozóra elhelyezhető – *és betöltés után működőképes!* – operációs rendszerek megjelenését, ilyen irányú fejlesztésekre a KNOPPIX disztribúciók mutatnak jó példát.

A Linux tehát *nem egy operációs rendszer*, sokkal inkább egy „operációs rendszer-család” vagy inkább számítástechnikai (jelen esetben a szó alatt a számítógép kezelésének képességét értve) *filozófia*.

Disztribúciók (http://www.linux.org/dist/list.html):

3-2. ábra: Elterjedtebb Linux disztribúciók

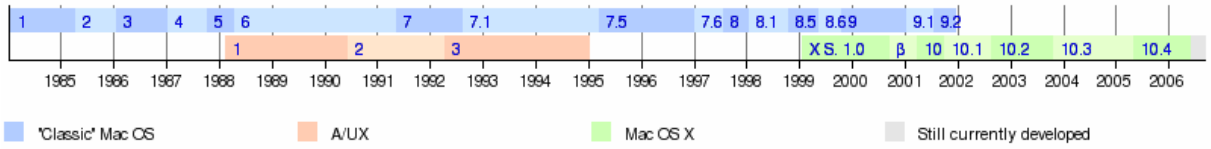
3.1.1.3. Az APPLE

A **Mac OS X** az Apple Computer 2001-ben megjelent és azóta is folyamatosan továbbfejlesztett grafikus operációs rendszere. Szakítva a korábbi Mac OS-ek felépítésével a tízes rendszert Unix alapokra helyezték, és a forráskód nagy részét a Darwin Project keretein belül nyílttá tették. A teljes váltás hátránya volt, hogy a korábbi rendszerek alá írt programokat csak a *Classic* nevű emulátorban képes futtatni; előnyeként viszont azt

említhetjük, hogy modern általános felhasználású operációs rendszert kaptak végeredményként.

Történet

A Mac OS X története 1985-ben kezdődött, mikor is Steve Jobs-ot eltávolították az Apple éléről. Ekkor alapította meg Jobs a *NeXT Computer*t, amelynek NeXTstep - később OpenStep - operációs rendszere a későbbi OS X alapjává vált. Jobs NeXT-es évei alatt az Apple is próbálkozott új operációs rendszer fejlesztésével, de a tervezetek sorban hamvába haltak, a várva várt Copland-ből sosem lett végleges kiadás, bár elemei beszivárogtak a Mac OS 8-ba. Hiába, a Windows 10 év után felzárkózni látszott, leginkább a védett memória és modern multitasking terén, melyek hiányoztak a Mac OS-ből. A cég ezért operációs-rendszer keresésbe fogott; a BeOS és a Windows NT kizárása után, 1996-ban végül 400 millió dollárért megvásárolta a NeXT Computert. Ezzel a lépéssel az Apple-höz került az objektum-orientált, UNIX-alapú OpenStep, valamint vezető fejlesztője, Avie Tevanian. Talán jelentősebb, hogy egy időben hazatért az alapító Steve Jobs, aki hónapokon belül visszavette a cég irányítását. A Mac OS X fejlesztését az OpenStep, Mac OS 9 és BSD alapjain kezdték el a *Rhapsody* projekt keretein belül. E fejlesztés végállomása a Mac OS X 10.0 2001-es nyilvános kiadása volt.



3-3. ábra: A MAC OS fejlődése

Verziók

A rendszer főbb változatai hagyományosan nagymacskákról kapják (kód)nevüket, melyek a 10.2-es verzió óta hivatalos névként is szolgálnak, a dobozon és marketingben is szerepelnek.

- 2001. március 24.-én jelent meg a 10.0-s verzió, azaz *Cheetah*: az első verzió lassú és kidolgozatlan volt, nagyon kevés alkalmazás ált rendelkezésre a felhasználók számára.
- 2001. szeptember 25.-én a 10.1-es *Puma*: a fejlesztés eme fázisában pótoltak néhány eddigi hiányosságot, mint pl. a DVD kezelés.
- 2002. augusztus 24.-én a 10.2-es *Jaguar*: nagyon jelentős fejlesztéseket (több mint 150) hozott: mint pl. MS Windows hálózati támogatás, Quartz Extreme: közvetlen videókártya használat (AGP), SPAM levél szűrés, konferencia beszélgetés, iChat csevegés az AOL Instant Messengerrel, tucatnyi új Apple Universal Access lehetőség, Sherlock 3: Web services, új Unix alapú nyomtató rendszer, stb.
- 2003. október 24.-én a 10.3-as *Panther*: az előző verzióban bevezetett újdonságok hibáit kijavították, s tovább fejlesztették: a Finder új felhasználói interfészt kapott, gyorsabb lett a keresés és testreszabható a SideBar, az Expose lett az új ablakkezelő, gyors felhasználó váltás (FUS) anélkül, hogy a felhasználónak ki kellene lépnie, iChat bővül a videókonferencia lehetőséggel, PDF állomány olvasása, stb.
- 2005. április 29.-én a 10.4-es, *Tiger* jelzésű aktuális főverzió: több, mint 200 új tulajdonság: Spotlight: új meta adatokon alapuló file-kereső rendszer, Dashboard: kicsi futó alkalmazások (Widgets) futtatása az asztalon, Smart Folders: Virtuális

könyvtár a felhasználói igény szerinti tartalommal, új iChat többszereplős videó konferenciák lebonyolításának lehetősége, Core Image és Core Video: valósidejű kép és videó „szerkesztés”, 64 bites memória támogatás a G5-s programok számára, UNIX programok aktualizálása, teljesen új API Core Data a Cocoa programok számára.

- 2006. végére várható a 10.5-ös változat, kódnevén *Leopard*, az Apple állítása szerint a Leopard támogatja majd mind a PowerPC[16], mind az Intel alapú Macintosh gépeket.

Minden Mac OS X rendszernek van Server kiadása is, amely munkacsoport kezelési lehetőséggel, hálózati- és adminisztratív szolgáltatásokkal bővíti az asztali kiadást. A Server kiadás drágább, de legfrissebb verziója jár minden Xserve kiszolgálóhoz. Ma már elmondhatjuk az OS X-ről, hogy megjelent rá minden olyan jelentős program, amely azelőtt az OS 9-en létezett. Sőt, a Mac OS X *Aqua* felületkezelője mellett párhuzamosan futtatható hagyományos X11-es ablakkezelő is, így a UNIX-okra utolérhető rengeteg program közül is válogathat a felhasználó.

Felhasználói élmény

Már az első bekapcsoláskor feltűnik, hogy az OS X felülete eltér a Windows alatt megszokottól. Alapértelmezetten a képernyő alján megtaláljuk a **dokk**ot, ami érdekes keveréke a már futó és az épp nem futó, de gyakran használt alkalmazásoknak. A képernyő tetejét pedig a fix **menüsor** foglalja el, amin az épp fókuszban levő alkalmazás beállításait láthatjuk. Vannak továbbá menüsorban futó alkalmazások is, továbbá egyes programok visszajelző funkcióját is meg lehet itt jeleníteni (óra, hálózat, cpu terhelés, stb). Itt kap helyet a Tiger rendszerben bemutatott SpotLight kereső kezelőfelülete is.

A Unix-szerű rendszereknél megszokott módon a felhasználó számára egy felhasználói könyvtárat, a Home-ot, ahol teljes jogkörrel rendelkezik, hasonló jogokat élvez a felhasználó az általa létrehozott könyvtárakban is. Ezzel szemben a rendszerkönyvtárakhoz adminisztrátori jogosultság szükséges.

Feltűnő különbség továbbá, hogy a legtöbb alkalmazás egy .app fájlnak látszik, aminek bár megtekinthetjük tartalmát, a mindennapi használatban erre nincs szükségünk. Ez az „Application bundle”-szerkezet az OS X egyik elődjéből a NeXTstep operációs rendszerből származik, ahonnan a rendszer a Mach mikrokernelt is örökölte. Ennek köszönhetően keveset kell válogatnunk az Applications könyvtárban, ha futtatható fájlt akarunk találni, hiszen minden .app-ot el tudunk indítani egy dupla kattintással.

Fontosabb tulajdonságok

- Quartz és Quartz Extreme ablakkezelő rendszer, PDF alapú megjelenítés
- OpenGL ablak elemek megjelenítése a képernyőn közvetlen hardver eléréssel
- 256*256 pixel méretig skálázható 32 bites színmélységű ikonok
- Unicode alapú karakterkezelés
- Egyszerűen állítható a rendszer nyelve (2005 augusztustól letölthető magyar lokalizációval)

---

[16] A **PowerPC** az Apple, az IBM és a Motorola összefogásából született RISC processzor architektúra.

- Beépített monitorkalibrációs eszközök (grafikai és nyomdai felhasználás támogatása)
- Gyors ablakváltás, asztal elérés az Exposé-val (10.3 Panther óta)
- Védett 128-bites kulcsú fájlkezelés (FileVault) a felhasználók HOME könyvtárában
- SpotLight keresőtechnika rendszerbe integrálása (10.4 Tiger óta)
- Dashboard segítségével apró kiegészítő szerkentyűk (widgetek) futtatása az asztalon (10.4 Tiger óta)
- SMART Folders lehetővé teszi a könyvtárak dinamikus kezelését a megadott szempontok szerint.
- Beépített szinkronizáló, amely kezeli a felhasználói adatok (naptár, találkozók, telefonkönyv) változását egy központi adatbázison keresztül.
- Apache, ftp-szerver integráció

Látható tehát, hogy a UNIX programrendszer „családfája" elég szerteágazó. A továbbiakban (ahol lehet, az általánosság talaján maradva) bemutatjuk a UNIX/Linux rendszereknek a felhasználói munkavégzés szempontjából fontosabb jellemzőit.

3.1.2. A UNIX/Linux operációs rendszerek legfontosabb jellemzői

A UNIX a valaha készített legerősebb valósidejű erőforrás megosztáson alapuló operációs rendszer. A Unix operációs rendszert úgy tervezték, hogy számos felhasználó képes ugyanabban az időben használni egy gépet, osztozva a gép erőforrásain. Az operációs rendszer összehangolja a használatban lévő erőforrásokat, megengedve egy személynek, például, hogy futtasson egy helyesírás ellenőrző programot, mialatt egy dokumentumot hoz létre, míg valaki más szerkeszti, formázza, grafikát készít, mindezt egy időben függetlenül a másik felhasználótól.

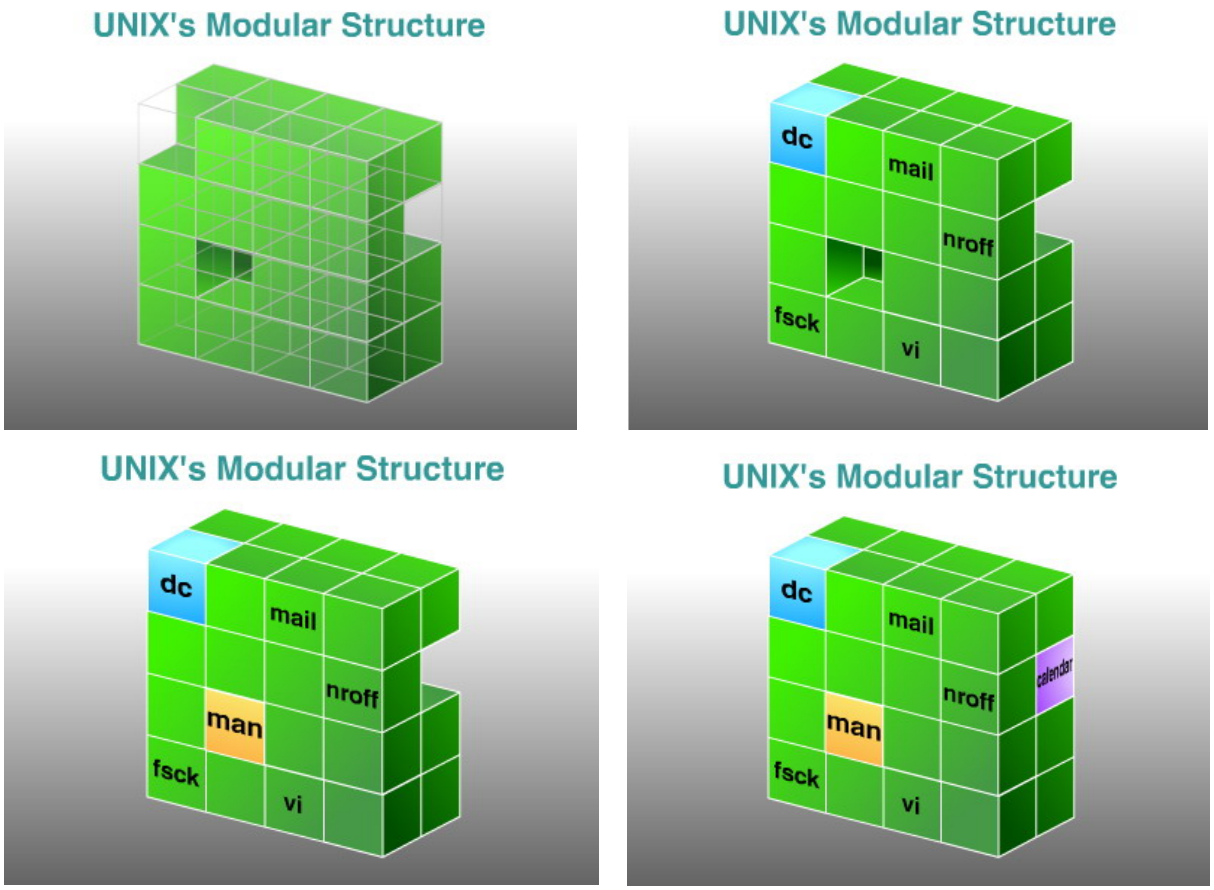
A UNIX-t programozók tervezték (programozók számára), a fejlesztési környezet nagyon erős és rugalmas. A rendszer megtalálható az üzleti világban, tudományban és iparban. Számos telekommunikációs és adatátviteli rendszer is UNIX-t használ a folyamatok adminisztrálására és figyelésére.

Alapvetően a rendszert közepes méretű számítógépekre tervezték, de hamarosan átvitték nagyobb, sokkal erősebb nagyszámítógépekre is. A személyi számítógépek népszerűségének növekedésével készültek UNIX változatok ezekre a géptípusokra is.

A UNIX legfőbb tulajdonságai a következők:

- többfeladatos képesség, párhuzamos programfuttatás;
- több-felhasználó képesség;
- átvihetőség: a UNIX egyik legfontosabb tulajdonsága a hordozhatóság, ami annyit jelent, hogy eltérő rendszerű számítógépeken képes futni, a kód minimális változtatásával. Az operációs rendszer frissítésekor a felhasználók adatai teljes egészében megmaradnak. A UNIX új verziói lefelé teljesen kompatibilisak, megkönnyítve a felhasználók áttérését az új verzióra.
- Unix programok: nagyszámú, az operációs rendszerrel együtt települő vagy telepíthető kiegészítő illetve segédprogram. Ezeket alapvetően két nagy csoportra oszthatjuk: a „nélkülözhetetlen" programokra (Integral utilities), melyek az operációs rendszer működéséhez elengedhetetlenül szükségesek, mint például a

parancsértelmező (interpreter), és a „felhasználói” programokra (Tools), melyek gondoskodnak a felhasználók igényeinek kiszolgálásáról (pl. levelezés, szövegszerkesztés, hálózati munka támogatása, ld. *3-4. ábra*).

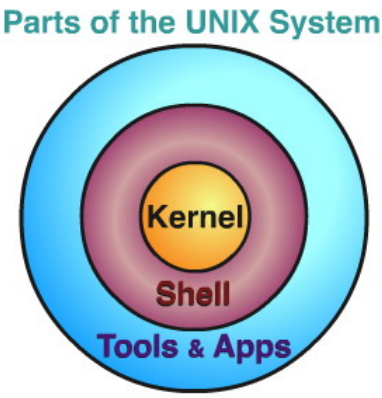


3-4. ábra: A UNIX rendszer moduláris felépítése

A UNIX felépítése

A UNIX rendszer 3 szintű funkcionális részből áll:

- a **kernel** – a rendszer magja –, amely ütemezi a feladatokat és gazdálkodik a tárolókkal,
- a **shell** – héj –, amely kapcsolatot tart és értelmezi a felhasználó utasításait, programokat indít el, és végrehajtja őket,
- és azok az **eszközök és felhasználói programok**, melyek az operációs rendszer szolgáltatásait bővítik.



3-5. ábra: A UNIX rendszer felépítése

A kernel az operációs rendszer szíve, felügyeli a hardvert és a szükségleteknek megfelelően be vagy kikapcsolja a rendszer számára. Ha megkérdezzük a számítógépet a rendelkezésre álló tárolókról, a kernel megmondja a rendszerben található valamennyi állomány és könyvtár adatát és a kijelzőre továbbítja. A shell („héj”) mindenkor kapcsolatot teremt a felhasználó és a számítógép között, értelmezve és végrehajtva a felhasználó utasításait. A rendelkezésre álló shell programok többnyire a parancsértelmezők (általában Burne és/vagy C shell alapúak, menü vezérlésűek). A shell egyik fontos tulajdonsága, hogy képes összekapcsolni („pipe”, „csatorna”) több utasítást azáltal, hogy az egyik parancs kimenetét a következő utasításnak bemenő adatként adja át. A rendszer legkülső szintjén helyezkednek el azok a programok, amelyek a felhasználói igények kiszolgálására készültek („tools and applications”, *3-5. ábra)*. A UNIX rendszerben (is) programok százai állnak rendelkezésre kezdve a szövegszerkesztőktől, az üzleti alkalmazásokon keresztül, a programozásig. A programok egy részét független fejlesztő cégek készítik.

3.1.2.1. Alapfogalmak

A Linux alapú rendszerek működési sajátosságainak áttekintéséhez elengedhetetlen, hogy bizonyos fogalmakat (esetleg módszereket) előre definiáljunk. Olyan fogalmakat, amelyek ugyan általában léteznek, ismertek más operációs rendszerek alatt is, azonban a Linux a megszokotthoz képest más (pl. kibővített) értelemben használja.

account (felhasználói fiók, egyes magyar forrásokban szokás „témaszám”-nak is fordítani): a Linuxban alapfogalom. Tekintve, hogy eredendően többfelhasználós rendszerről van szó, Linux alapú rendszert csak felhasználói azonosítás után tudunk használni. Az egyes felhasználókat egyértelműen megkülönböztető logikai leírás (azonosító) az account. (A szó eredetileg számlát jelent, a kifejezés abból ered, hogy egyes operációs rendszerek alkalmazásakor a felhasznált processzoridőért, tárolóterületért pénzt számoltak/számolnak fel.)

UID (user identifier, röviden user ID): a felhasználó rendszerszintű azonosítója, az „account” fizikai megjelenési formája. A Linux a rendszerben minden egyes felhasználónak egyedi azonosítója van, ami azonban nem a felhasználó számára megjelenő szöveges információ (a felhasználó neve), hanem egy numerikus „kód”.

GID (group identifier, group ID): csoportazonosító. A Linux alapú rendszerekben (mint minden többfelhasználós/hálózati operációs rendszerben) léteznek csoportok, amelyek a felhasználók kezelésének egyszerűsítését támogató/lehetővé tevő logikai kategóriák. Minden felhasználó be van osztva egy (vagy több) csoportba, a GID annak a csoportnak az

azonosítója, amelybe a felhasználó tartozik. A Linux alapú rendszerek a csoportokat az állományokkal kapcsolatos (hozzáférési, jogosultsági) műveletekben alkalmazzák.

EUID (*effektiv user ID*): általában egyenlő az UID-del, de bizonyos esetekben (pl. a jogosultsági rendszer „set-uid-bit"-jei – *ld. később!*) más is lehet. Gyakorlati jelentősége abban áll, hogy amíg hagyományos esetben minden tevékenység az adott feladatot elindító felhasználó jogosultságával rendelkezik, addig ilyen módon egy adott folyamatnak több jogot lehet adni, mint ami a folyamatot elindító felhasználónak van.

EGID (*effektiv group id*): ugyanaz, mint EUID, csak a csoport-azonosítóra.

root: kitüntetett szerepű felhasználó, alapértelmezés szerint a rendszer valamennyi erőforrására vonatkozóan minden joggal rendelkezik (*rendszergazda, szuperfelhasználó, superuser, supervisor*).

process (job, folyamat, tevékenység): a Linux másik alapfogalma. A Linux filozófiája szerint a rendszerben végrehajtódó minden tevékenység egy-egy folyamatként valósul meg, ilyen értelemben a process a legkisebb párhuzamos feldolgozásra alkalmas egység. Minden elindított parancs vagy program egy vagy több process-t határoz meg. A process-ek között hierarchikus viszony áll(hat) fenn: egy futó folyamat elindíthat egy újabb folyamatot, ebben az értelemben az elsőt szülő-, a másodikat gyermek- (vagy al-) processnek nevezzük. Szabályos működés mellett egy szülő-process csak abban az esetben érhet véget, ha előbb minden gyermek-process-e szabályosan befejeződött (a gyermek-process-ek befejeződésüket jelzik a szülőnek). A process-ek megkülönböztetése azonosítószámmal történik.

PID (process ID): folyamat-azonosító. A felhasználókhoz hasonlóan az operációs rendszer valamennyi erőforrás-használója (*így a folyamatok is!*) egyértelmű azonosítóval (tulajdonképpen egy sorszám) rendelkeznek.

multitaszk: több feladat egyidejű végrehajtását jelenti. Egy processzorral rendelkező számítógépeken az egyidejű végrehajtás csak látszólagos, hiszen a processzor csak egy feladattal tud foglalkozni egyszerre (*ld. folyamatkezelés!*). A legkisebb egység amely párhuzamos feldolgozásra kerülhet a process. A feladatok váltogatását az ütemező végzi, amely különböző stratégiák szerint dolgozhat (*ld. folyamatkezelés!*). A Linux prioritási szinteket használ (az egyes prioritási szinteken alapértelmezés szerint más és más tevékenységek végezhetők el), de lehetővé teszi azt is, hogy a felhasználó megváltoztassa a saját process-einek a prioritását.

A Linux *preemptív multitaszkos* operációs rendszer, ami azt jelenti, hogy amikor egy adott folyamat számára kijelölt időszak letelt, akkor a kernel megszakítja a folyamat futását, és másik folyamatnak adja át a vezérlést. Az operációs rendszer nem teszi lehetővé, hogy egy folyamat a végtelenségig magánál tartsa a vezérlést, és így megakadályozza a többi folyamat futását. Azonban a prioritási soron belül lehetőség van a szokásos körbenjáró (*round robin, ld. folyamatkezelés*) ütemezés helyett sorrendi (*FIFO, ld. folyamat-kezelés*) ütemezés kérésére is. Ezáltal „szoft-real-time"-nak nevezett, lényegében virtuális valós idejű ütemezést is meg lehet valósítani. Ilyenkor a rendszer nem veszi el a futás jogát a processztől, csak ha az lemond róla. Azonban a Linux nem real-time operációs rendszer (bár van ilyen irányú fejlesztései is), és ez azt jelenti, hogy több futó folyamat esetén bizonyos időközönként mindegyikre rákerül a vezérlés, azonban a két aktív állapot között eltelt időre nincs szigorú időkorlát.

multiuser: több felhasználó egyidejű kiszolgálását jelenti. A Linux esetében ez elsősorban nem kifejezetten fájlok megosztását jelenti, hanem inkább több program virtuális párhuzamos futtatásának a képességét – akár egy felhasználó folyamatai esetén is, de több felhasználóra is vonatkozhat. Ez utóbbi esetben egy számítógépre gépre több ember jelentkezhet be egyszerre (azonos időben), és egyszerre tudnak dolgozni anélkül, hogy

zavarnák egymás munkáját. (Ez természetesen feltételezi azt, hogy a rendszernek meg kell tudnia különböztetni egymástól a felhasználókat – *ld. felhasználói fiókok*). Minden felhasználóhoz előre definiált jogok és engedélyek tartoznak, amelyeket a rendszer mindig ellenőriz, amikor a felhasználó szeretne hozzáférni valamihez. Ez szükséges az erőforrások megfelelő szétosztásának, és a biztonságnak az érdekében.

a memóriakezelés sajátosságai: a modern operációs rendszerek képesek arra, hogy látszólag több memóriát biztosítsanak a programoknak, mint amennyi fizikailag a rendelkezésükre áll (*ld. memóriakezelés*). Annak érdekében, hogy a merevlemezt virtuális memóriakezelésre használni lehessen, a Linux egy külön ilyen célra fenntartott tárterületet (*swapfájl, swap-partíció*) kezel. Linux alatt ennek mérete dinamikusan, működés közben is változtatható, tehát az operációs rendszer leállítása nélkül lehetőségünk van a virtuális memória átméretezésére. Érdekesség, hogy alapértelmezés szerint egy swap-partíció mérete maximum 128 MB lehet, de használhatunk belőle többet is (maximum 16 darabot), gyakorlatilag a swap tényleges mérete célszerűen a rendelkezésre álló fizikai memória kétszerese.

gyorsítótár (*buffer cache*): a memória-kezeléshez szorosan kapcsolódó fogalom, a memória (működési szempontból) speciálisan kezelt része, a Linux rendszerek lemezeléréshez használt gyorsítási technikája, amelyet a kernel kezel. Működési elve megegyezik a klasszikus gyorsítótár-kezelési technikával: a lemezre írandó anyag is először a gyorsítótárba kerül, és vagy egy megadott idő elteltével (pl. 30 másodperc) íródik ki a lemezre, vagy pedig akkor, ha a rendszer számára elegendő mennyiségű anyag összegyűlik a gyorsítótárban. *Ezért fontos, hogy ne kapcsoljuk simán ki a számítógépet, hanem mindig szabályosan állítsuk le a rendszert a megfelelő parancsokkal (ld. később)!* A Linux alatt a gyorsítótár mérete is dinamikusan változik a rendszer terhelésétől függően – szélsőséges esetben akár az éppen szabad fizikai memória teljes egészét is erre a célra használja.

demand paging (igény szerinti lapozás, *ld. memória-kezelés*): szintén a memória-kezeléshez kötődő teljesítményt növelő módszer. Azt jelenti, hogy egy futtatható fájl végrehajtásakor nem az egész fájl töltődik be a memóriába, hanem mindig csak azok a lapjai, amikre a végrehajtás során éppen szükség van. Mivel minden programnak vannak olyan részei melyek csak egyszer (vagy akár egyszer sem) futnak le, ezeket a részeket vagy be sem tölti a rendszer, vagy miután lefutottak felszabadítja az általuk elfoglalt memóriaterületet.

osztott kódkönyvtárak: használatának alapelve az, hogy a programok C nyelven íródnak, és valószínűleg sokban van olyan függvény, amely más programokban is előfordul. Ezeket felesleges lenne minden programmal a memóriába tölteni, elég egyszer, és meg kell mondani a programoknak, hogy hol keressék ezeket a függvényeket a memóriában. Ezt csinálja a *dinamikus linker*, amely a programokba beépített programrészletnek segítve gondoskodik a függvények megtalálásáról, illetve a memóriába töltésükről, amennyiben még nem lennének betöltve.

kernel: az operációs rendszer magja, alapértelmezés szerint az éppen futó Linux disztribúció (*ld. később*). Szerepe a felhasználó, a programok és a hardver eszközök közötti kapcsolattartás. Memóriaterületet biztosít az összes futó process számára, ezenkívül egységes felületet nyújt a programoknak, amelyen keresztül kommunikálhatnak a hardverrel. (Alapértelmezés szerint egy Linux alapú rendszerben közvetlen módon csak a kernel szolgáltatásai érhetik el a hardver egységeket.)

kernel frissítés: az operációs rendszer újabb szolgáltatásokkal történő kiegészítése, illetve a meglevő szolgáltatások javítása. Az újabb kerneleknek általában több eszközmeghajtójuk van, általában jobb a process-kezelésük, gyorsabb vagy stabilabb, és javítja a korábbi verzió ismert hibáit.

eszközök: A Linux rendszerek *mindent fájlként kezelnek:* a merevlemezeket, a terminálokat, az audio-eszközöket, meghajtókat, stb. Az elv lényege, hogy ilyen módon az eszközöket (bármilyen eszközről – perifériáról – legyen is szó!) a fájlokhoz hasonló módon tudják elérni. A „/dev" könyvtárban találunk meg minden fájlt, ami az eszközökhöz tartozik. Az eszközöknek két fajtájuk van: karakter- és blokk-orientált eszközök. *Karakterorientált* eszköz például a terminál, a soros port. *Blokkorientált* eszközök például az adattároló eszközök. Az eszközöket két szám jellemzi: a fő- és az al-eszközszám. A főeszközszám adja meg az eszköz típusát. Ha ugyanabból a típusból több eszköz is van, akkor ezeket az aleszközszám különbözteti meg egymástól.

démonok: speciális processzek, amelyek a háttérben (a felhasználó elől „rejtetten") futnak, párhuzamosan más programokkal. Az operációs rendszer nagy egységei önálló programként így futnak. Konfigurációjuk módosítása esetén anélkül újraindíthatóak, hogy magát az operációs rendszert is újra kellene indítani. Jellemző példák: nyomtató démon: lpd; rendszernaplózás: syslogd; időzítési feladatok: cron, at; Internet démon: inetd; fájlrendszer-kezelés: rpc.nfsd. (Fontos megjegyezni, hogy az egyes szolgáltatásokhoz egy jól meghatározott *port-szám* tartozik, amelyeken keresztül el lehet érni az adott szolgáltatást: a rendszer külső védelme szempontjából ennek igen nagy jelentősége van!)

shell: a Linux alapú rendszerekben (a hagyományos értelemben vett felhasználói felülettől különböző értelemben!) a rendszer parancsértelmező komponense. (Azon sajátossága miatt, mely szerint ugyanazon operációs rendszer több különböző parancsértelmezőt is képes kezelni, egyes források nem is tekintik az operációs rendszer szerves részének!) Minden felhasználó bejelentkezésekor egy parancsértelmező indul el. A parancsértelmező szabványos bemenete és kimenete a terminál. Egy *promptot* jelenít meg (ami egyénileg beállítható), jelezve, hogy készen áll a feladatok végrehajtására. Működése a process-hierarchia szerint valósul meg: ha a felhasználó elindít egy parancsot, akkor a parancsértelmező elindít egy gyermek-process-t, ami lefuttatja a kért parancsot. A gyermek-process futása közben a parancsértelmező annak a megszűnésére vár. A gyermek-process megszűnésekor a parancsértelmező újra megjeleníti a promptot.

konzol: (a Linux alapértelmezése szerint) a számítógépes rendszernek nem része a monitor és billentyűzet, csak hozzá lehet csatlakoztatni, éppen ezért szükség van egy kommunikációs felületre – ez a konzol: a számítógép és a felhasználó közötti kommunikáció felülete. Alapértelmezés szerint ide érkeznek azok az üzenetek, amiket a rendszer küld, biztosítja a parancskiadást (*ld. parancsértelmező*) és létezik grafikus felülete is (xconsole). A Linux lehetővé teszi, hogy a fizikailag egyetlen számítógépet virtuálisan úgy használjuk, mintha több különböző számítógépen (terminálon) dolgoznánk. Természetesen egyszerre csak egy terminálon tudunk dolgozni, de az elindított programok párhuzamosan futnak egymással. Az egyes *virtuális konzolok* között váltogathatunk az Alt-Fx (x=1...12) billentyűkombinációval.

fájlrendszer: a fájlrendszer a lemezen tárolt adatok kezelhetőségét biztosítja. Annak érdekében, hogy a Linux más fájlrendszerek támogatását is tudja biztosítani, a kernel és a fájlrendszerek között létezik egy szint, amelyet *virtuális fájlrendszernek* neveznek. Ez rendelkezik azokkal a rutinokkal, amelyek szükségesek egy fájlrendszeren történő műveletvégzéshez. Ez biztosítja a különböző fájlrendszerek közötti átjárhatóságot, mert a felhasználónak nem is kell tudnia, hogy milyen fájlrendszeren történik a műveletvégzés, csak kiadja a parancsot, és az érintett fájlrendszer-kezelő lefordítja a megfelelő fájlrendszer-hívásokra.

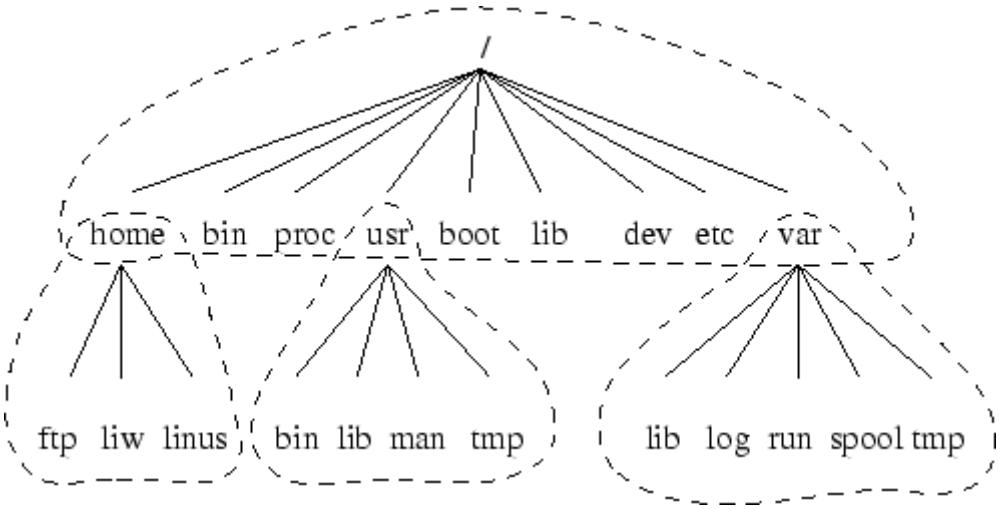
(hozzáférési) jogosultságok: a fájlokhoz tartozó olyan jogosultságok, amelyek meghatározzák, hogy melyik felhasználó melyik fájlon milyen műveletet hajthat végre.

Az alapfogalmak rövid áttekintése után nézzük meg a felhasználói szempontból legfontosabb tevékenységek támogatásával kapcsolatos komponenseket és azok kezelésének estközrendszerét!

3.1.3. Fájlkezelési és nyomtatási rendszerek

A UNIX/Linux alapú operációs rendszerek fájlrendszere egy elég összetett és szigorú szabályok szerint kialakított tárolási rendszer. A teljes könyvtárfa struktúrát (*3-6. ábra*) úgy tervezték, hogy azt kisebb részekre lehessen bontani, amelyek külön lemezpartíción helyezkednek el, hogy a korlátos lemezméreteket összegyűjthessük és könnyebb legyen a biztonsági mentés valamint az egyéb rendszeradminisztráció. A legfontosabb részek: a gyökér („root”, „/”), „/usr”, „/var” és a „/home” fájlrendszerek. Mindegyik résznek különböző a feladata. A könyvtárfát úgy tervezték, hogy jól működjön hálózatba kötött Linuxos gépek esetében is, amikor egyes részek csak olvasható módon kerülnek megosztásra, pl. CD-ROM-ról.

A Linux fájlrendszer ismertetésénél természetesen nem lehet cél minden fájl részletes leírása, csak a rendszer áttekintése a fájlrendszer oldaláról.



3-6. ábra: a Linux fájlrendszere

Az egyes részek szerepét illetve leírását (a teljesség igénye nélkül) az alábbiakban foglalhatjuk össze:

/ (a gyökér fájlrendszer)

A gyökér fájlrendszer minden gépnél egyedi kell, hogy legyen, és tartalmaznia kell mindazt, ami a „boot”-oláshoz kell addig a pillanatig, amikor már a többi fájlrendszer is beilleszthető („mount”-olható). Ez a fájlrendszer általában helyi lemezen helyezkedik el, ezért a gyökér fájlrendszer tartalma csak az egyfelhasználós (single user) üzemmódhoz elegendő. Ez a rendszer tartalmazza az elrontott rendszer javításához és az elveszett fájlok biztonsági mentésből való visszaállításához szükséges eszközöket is.

A gyökér fájlrendszernek általában kisméretűnek kell lennie, mivel a kritikus fájlokat tartalmazza, és egy kicsi, ritkán módosított fájlrendszernek jobb esélye van a sérülések elkerülésére. Egy sérült gyökér fájlrendszer általában azt jelenti, hogy a rendszer nem képes boot-olni, csak különleges segítséggel (pl. floppy-ról), és ezt nem érdemes kockáztatni.

A gyökérkönyvtár általában nem tartalmaz fájlokat, legfeljebb a rendszer standard „boot image fájl”-ját, melynek szokásos neve /vmlinuz. A további fájlok a következő könyvtárakban vannak:

/bin (/sbin): a boot-olás során szükséges parancsok, melyeket (a boot-olás után) a felhasználók is használni fognak. Az „/sbin” könyvtár szerepe ugyanaz, mint „/bin” könyvtáré, de az ebben levő parancsokat nem a „normál” felhasználóknak szánták (habár néha szükségük lehet rá, és használhatják is, ha van rá engedélyük).

/etc: gép-specifikus konfigurációs fájlok.

/root: a rendszergazda (kiemelt felhasználó, „root”) saját („home”) könyvtára.

/lib: megosztott könyvtárak a gyökér fájlrendszer programjaihoz.

/lib/modules: betölthető kernel modulok, különösen azok, melyek a rendszer felállásához (boot-oláshoz) szükségesek pl. egy baleset utáni helyreállításkor (pl. hálózati és fájlrendszer vezérlők).

/dev: eszköz-vezérlő fájlok.

/tmp: ideiglenes fájlok. A boot-olás után futó programoknak inkább a /var/tmp használata ajánlott a „/tmp” helyett, mivel az előző valószínűleg egy kisebb kapacitású lemezen van.

/boot: a boot betöltő („*bootstrap loader*”, mint pl. a LILO) által használt fájlok. A kernel image fájlok is gyakran itt vannak, nem a gyökérkönyvtárban. Ha több kernel image is van, a könyvtár gyorsan megnőhet, ezért jobb lehet külön fájlrendszerre rakni. Egy másik ok erre az, hogy így bebiztosíthatjuk, hogy a kernel image fájlok az IDE lemez első 1024 cilinderén vannak.

/mnt: csatlakozási pontok az ideiglenes csatlakozásokhoz. A programok nem várják el, hogy egy fájlrendszer legyen automatikusan csatlakoztatva a „/mnt” alá. A „/mnt” felosztható egyéb alkönyvtárakra, pl. „/mnt/dosa” lehet az MS-DOS formátumú floppik csatlakozási helye.

”/usr” fájlrendszer

A ”/usr” fájlrendszer tartalmazza a parancsokat, könyvtárakat, kézikönyv lapokat, és egyéb, a normál üzem alatt nem változó fájlokat. A ”/usr”-beli fájlok nem specifikusak egyetlen gépre vonatkozóan sem. Ezek miatt lehetőség van a fájlok hálózaton keresztüli megosztására, ami nagyon takarékos, hisz sok lemezterület megtakarítható így. (Könnyen lehet több száz megabájtnyi a ”/usr” tartalma.) További előny, hogy könnyebb lesz a rendszer karbantartása, mert csak a központi ”/usr” partíción kell az alkalmazásokat frissíteni, nem minden gépen külön-külön. De még helyi lemez esetén is megtehetjük, hogy a ”/usr”-t csak olvasható módon csatlakoztatjuk, ami csökkenti a fájlrendszer sérülésének veszélyét egy rendszerösszeomlás esetén.

A ”/usr” fájlrendszer gyakran nagyon nagy, mert lényegében minden program ide kerül az installáláskor. A ”/usr” fájljai rendszerint egy Linux disztribúcióból származnak; a helyileg installált programok és egyebek a ”/usr/local”-ba kerülnek. Ez lehetővé teszi, hogy a rendszert a terjesztés egy új változatával frissítsük, vagy akár egy teljesen új terjesztéssel, anélkül, hogy mindent újra kellene installálni. A ”/usr” néhány alkönyvtárát az alábbiakban röviden ismertetjük.

/usr/X11R6: az XWindow System minden fájlja. Az X fejlesztésének és installálásának egyszerűsítése érdekében az X-hez tartozó fájlok nem integrálódnak be a rendszer egyéb részei közé. A ”/usr/X11R6” alatt található könyvtárak szerkezete a ”/usr” felépítéséhez hasonló.

/usr/bin: majdnem minden felhasználói parancs itt van (néhány a ”/bin” és a ”/usr/local/bin”-ben is található).

/usr/sbin: rendszeradminisztrációs programok, melyek nem szükségesek a gyökér fájlrendszer számára, pl. a legtöbb kiszolgáló program.

/usr/man, /usr/info, /usr/doc: kézikönyv lapok, GNU dokumentumok és vegyes dokumentáció fájlok.

/usr/lib: programok és alrendszerek nem változó adatfájljai, többek között globális (system-wide) konfigurációs fájlok. A ”lib” név a ”library”-ból származik; eredetileg a programozási szubrutinok voltak elhelyezve itt.

/usr/local: a helyileg installált szoftverek és egyéb fájlok.

”/var” fájlrendszer

A ”/var” fájlrendszer változó fájlokat tartalmaz, mint pl. a ”spool” könyvtárak (a levelezéshez, hírekhez, nyomtatáshoz), naplófájlokat (”log file”), formázott kézikönyvlapokat és ideiglenes fájlokat. A hagyományok szerint minden, ami a ”/var”-ban van, kerülhetne akár valahova a ”/usr”-be is, de akkor nem lehetne a ”/usr”-t csak olvasható módban csatlakoztatni.

A ”/var” olyan adatokat tartalmaz, melyek a rendszer normális üzeme alatt megváltoznak. Ez rendszer specifikus, azaz nem osztható meg a hálózaton más számítógépekkel.

”/home” fájlrendszer

A ”/home” fájlrendszer tartalmazza a felhasználók ”home”-könyvtárait, azaz minden valós adatot a rendszerben. A ”home”-könyvtárak külön fájlrendszerbe való elkülönítése könnyebbé teszi a biztonsági mentéseket; a többi részt általában nem, vagy csak ritkábban kell elmenteni, mivel ritkábban változnak. Egy nagy ”/home” fájlrendszert esetleg érdemes néhány fájlrendszerre szétszedni, ami egy plusz elnevezési szint bevezetését követeli meg a ”/home” alatt, mint pl. ”/home/students” és ”/home/teachers”.

Itt kell megjegyeznünk, hogy bár a különböző részeket fájlrendszereknek neveztük az előbb, nem feltétlenül szükséges, hogy valóban külön fájlrendszereken legyenek. Egy kicsi, egy-felhasználós rendszeren akár egyetlen fájlrendszeren is tárolhatók, ami egyszerűbbé teszi a dolgokat. A könyvtárfa esetleg máshogy is feloszthatő fájlrendszerekre annak függvényében, hogy mekkora lemezeink vannak és hogy hogyan foglaljuk a helyet különféle célokra.

A Linux fájlrendszer szerkezete a fájlokat cél szerint csoportosítja, azaz pl. a parancsok egy helyen vannak, az adatfájlok máshol, a dokumentáció egy harmadik helyen, és így tovább. Egy alternatíva lehetne a fájlok csoportosítására, hogy melyik programhoz tartoznak. Az utóbbi megközelítéssel az a baj, hogy nehézzé tenné a fájlmegosztásokat (a programkönyvtárak gyakran tartalmaznak statikus, megosztható, változó és nem megosztható fájlokat egyaránt), és néha még a fájlok (mondjuk a kézikönyvlapok) megtalálása is nagyon bonyolult lenne.

”/etc” fájlrendszer

Az ”/etc” sok fájlt tartalmaz, általánosságban elég nehéz meghatározni a szerepét.

/etc/rc vagy /etc/rc.d vagy /etc/rc?.d: szkriptek, vagy szkripteket tartalmazó könyvtárak, melyeket induláskor, vagy futásszint váltáskor futtat a rendszer.

/etc/passwd, /etc/group, /etc/shadow: a felhasználók adatbázisa a következő mezőkkel: felhasználói név (username), valódi név (real name), ”home” könyvtár, titkosított jelszó és egyéb információk. A „/etc/group” hasonló a ”/etc/passwd” fájlhoz, de a csoportokat írja le, nem a felhasználókat. Az árnyék jelszófájl (shadow password file) azokon a gépeken, ahol az

árnyék jelszavakat kezelő programok installálásra kerültek. Az árnyékjelszavak használata esetén a jelszavak a ”/etc/passwd” helyett a ”/etc/shadow”-ban tárolódnak, amelyet csak a ”root” olvashat. Ez megnehezíti a jelszavak feltörését.

/etc/fstab: a fájlrendszerek szokásos csatlakozási pontjait és paramétereit adja meg. Itt találhatók a boot-oláskor automatikusan (mount -a-val) csatlakozó, a később csatlakoztatható fájlrendszerek és az automatikusan bekapcsolandó swap területek leírásai.

/etc/profile, /etc/csh.login, /etc/csh.cshrc

Ezeket a fájlokat a ”shell” hajtja végre bejelentkezéskor vagy induláskor. Ez lehetővé teszi a rendszeradminisztrátornak, hogy minden felhasználónak azonos beállításokat adjon.

”/dev” fájlrendszer

A ”/dev” könyvtár a speciális eszközfájlokat tartalmazza az összes eszköz számára. Az eszközfájlok elnevezése speciális megállapodások szerint megy; ezeket az eszközlistában találhatjuk meg. Az eszközfájlok az installációkor kerülnek létrehozásra, illetve később a ”/dev/MAKEDEV” szkript segítségével. A ”/dev/MAKEDEV.local” egy szkript, melyet a rendszeradminisztrátor ír, és amely csak helyi használatú eszközfájlokat vagy kötéseket ír le pl. olyan esetekben, amikor az eszközvezérlő nem szabványos.

”/proc” fájlrendszer

A ”/proc” fájlrendszer egy látszólagos fájlrendszer. Nem is lemezen létezik, hanem a kernel hozza létre a memóriában. Arra lehet használni, hogy a rendszerről (eredetileg a ”process”-ekről) adjon információkat. Minden ”process”-nek saját könyvtára van a ”/proc” alatt a ”process” azonosító számmal megegyező néven.

Megjegyzendő, hogy bár az előbbi fájlok egyszerűen olvasható szöveges fájlnak vannak szánva, néha a formátumuk olyan, hogy nehezen értelmezhetők. Sok olyan program van, amely gyakorlatilag csak ezeket olvassa, majd könnyebben emészthetővé teszi.

3.1.3.1. FHS (Filesystem Hierarchy Standard)

A fájlrendszer-felépítés szabványának fejlesztési folyamata 1993-ban kezdődött, azzal az erőfeszítéssel, hogy a Linux fájl- és könyvtár-struktúrája újraszerkesztett legyen. Az FSSTND, amelyet 1994. februárban adtak ki, egy olyan fájlrendszer-felépítési szabvány, amely speciálisan a Linux operációs rendszerre érvényes. A rákövetkező átdolgozott verziók 1994-95-ben kerültek kiadásra.

1995 tavaszán a fejlesztés célja az volt, hogy az FSSTND egy még átfogóbb verzióját írják meg, amely nemcsak a Linux, hanem más UNIX alapú rendszerek által is elfogadott legyen. Eredményként egy összehangolt és összpontosított kiadás készült el, amely általánosan érvényes volt a UNIX típusú rendszerekre. A kiszélesedő alkalmazási terület felismerése a szabvány nevének megváltozását is maga után vonta, amely így Fájlrendszer-felépítési Szabvány (Filesystem Hierarchy Standard) lett, röviden FHS.

A Linux fájlrendszer jellemzői a hierarchikus felépítés, a fájl adatainak következetes kezelése és a fájl adatainak védelme.

E szabvány feltételezi azt, hogy az operációs rendszer alapját egy olyan FHS-alapú fájlrendszer alkotja, amely támogatja azokat a biztonsággal kapcsolatos lehetőségeket, amelyek a legtöbb UNIX* fájlrendszerben megtalálhatóak.

Definiálhatjuk a fájlok két világosan értelmezett kategóriáját: a megoszthatót a nem megoszthatóval szemben, és a változtathatót a nem változtatható tartalmú (statikus) ellenében. A megosztható adatokat több hálózati számítógép megosztottan használhatja fel, míg a nem

megosztható adatokat csak egy adott számítógép alkalmazhatja. Például a felhasználói ”Home”-könyvtárak lehetnek megosztottak, de az eszközmeghajtó lezáró fájljai (lock) nem. A statikus (nem változtatható tartalmú) fájlok közé tartoznak például a binárisok, a programkönyvtárak, a dokumentációk, és minden más, amely a rendszergazda beavatkozása nélkül nem változik meg.

A megosztható és a nem megosztható adatok közötti megkülönböztetés több okból is szükséges lehet, pl. hálózati környezetben (például, ha több számítógép alkot egy szervezetet) igen nagy mennyiségű adat az, amely megosztott az eltérő gépek között (adatterületet spórolva ezáltal, valamint megkönnyítve a fenntartás folyamatát), ugyanakkor bizonyos fájlok hálózati környezetben is olyan adatokat tartalmaznak, amelyek csak egy adott számítógéphez kapcsolódnak. Ezért ezeket a fájlrendszereket nem lehet megosztani (mértéktelenül semmiképpen sem).

A UNIX típusú fájlrendszerek történelmi implementációi egyazon fájlrendszerbe vegyítik a megosztható és a nem megosztható adatokat is, megnehezítve ezáltal a fájlrendszer széleskörű kiosztását.

3.1.3.2. Fájl-szintű jogosultsági rendszer

A Linux a felhasználókat három csoportra osztja, amikor a fájlokhoz és könyvtárakhoz való viszonyukat vizsgálja: a fájl tulajdonosa (**user**), csoport (**group**), egyéb (**others**).

A „csoport" segítségével csoportosíthatjuk a felhasználókat - lehetőleg - valamilyen logika szerint, hogy meg tudják osztani egymással a megfelelő fájljaikat vagy szabályozni tudjuk a hozzáféréseket.

A fájlokkal és könyvtárakkal három dolgot tehetnek a felhasználók: olvasás (**read**), írás (**write**), végrehajtás (**execute**)

Mindhárom felhasználói csoportra külön-külön be lehet állítani ezeket az engedélyeket. Fájloknál elég egyértelmű, hogy mit takar ez a három dolog: olvasás, írás és végrehajtás, de könyvtáraknál már nem az. Egy könyvtár olvasása azt jelenti, hogy egy erre a célra szolgáló paranccsal ki tudjuk listázni a könyvtár tartalmát. Egy könyvtárat akkor nevezünk írhatónak, ha a megfelelő parancsokkal bejegyzéseket tudunk létrehozni, módosítani vagy törölni az adott könyvtárban, illetve törölhetjük magát a könyvtárat. Talán a végrehajtás a legnehezebben megérthető, mert hasonlít egy kicsit az olvasás engedélyhez. De abban az esetben, ha egy könyvtárra csak olvasási jogunk van, akkor nem tudjuk megnézni a fájloknak a bejegyzéseit; míg ha végrehajtási jogunk van csak, akkor – ha ismerjük a fájl teljes nevét – az ls parancs megmutatja a fájlt (de csak azt az egy fájlt, mert a könyvtárban esetleg lévő többi fájlt már nem fogjuk látni), és el is tudjuk indítani (amennyiben az egy végrehajtható script vagy program, és van engedélyünk a fájlra).

Nézzük mindezt egy példán keresztül! Legyen a parancsunk egy egyszerű listázási parancs (ls –l valami), aminek a hatására megjelenő a képernyőn az alábbi sor jelenik meg:

-rwxr-xr-- 1 bela bela 6040 Jun 24 14:22 valami.

Az egyes részek és szimbólumok jelentése a következő:

- a legelső karakter (a példában a „-" jel) helyén a következő jelölések állhatnak:
  - -: közönséges fájl
  - b: blokkos eszköz (például: adattároló eszközök)
  - c: karakteres eszköz (például: nyomtató, terminál, stb.)
  - d: könyvtár
  - l: szimbolikus link

    - p: pipe (csatorna)
    - s: socket
- a következő kilenc karakter azt mutatja, hogy ki mit tehet a fájllal, a jogosultsági rendszernek megfelelő háromszor hármas csoportosításban (tulajdonos, csoport, egyéb):
    - r: olvasási engedély (read)
    - w: írási engedély (write)
    - x: végrehajtási engedély (execute)
    - -: az engedély hiánya (tiltás)
    - s: suid bit
    - t: sticky bit
- a fájlokhoz tartozó egyéb információk:
    - a fájlra mutató linkek száma
    - a fájl tulajdonosának azonosítója
    - a fájl csoporttulajdonosának azonosítója
    - a fájl mérete
    - időbélyeg(ek): a fájl létrehozásának, utolsó hozzáférésének (nem módosítási értelmű megnyitásának), utolsó módosításának dátuma
    - a fájl azonosítója (neve)

A speciális bitek jelentése:

A **suid** („Set User Identification”, felhasználói azonosító megváltoztatása) megértéséhez azt kell tudnunk, hogy időnként szükség van arra, hogy egy egyszerű felhasználó egy privilegizált felhasználó jogaival rendelkezzen. Talán a legegyszerűbb eset a jelszó megváltoztatása. Egy egyszerű felhasználó nem írhatja közvetlenül a rendszer jelszófájlját, hiszen akkor bármikor korlátlan jogokhoz juthatna, de a saját jelszavát meg kell tudnia változtatni. Ehhez viszont írnia kell a jelszófájlba. Ezt az ellentmondást oldják fel úgy, hogy a programot ruházzák fel privilegizált jogokkal, a suid bit beállításával.

A passwd parancs engedélyei a következők:

```
-rwsr-xr-x   1 root     root        28896 Jul 17  1998 /usr/bin/passwd
```

Látható, hogy a suid bit be van kapcsolva, így futásának idejére az őt futtató felhasználó rendszergazdai jogkört kap, tehát a root jogosultságaival olvassa és írja a /etc/passwd fájlt (mivel a root tulajdonában van a fájl).

A **sgid** („Set Group Identification”, csoportazonosító megváltoztatása) beállítása esetén a program annak a *csoportnak* a jogaival fog futni, akinek a fájl a birtokában van.

A **sticky bit** bekapcsolása fájlok esetén azt jelzi az operációs rendszernek, hogy a fájlt tartsa a memóriában a végrehajtás után is. Ennek a tulajdonságnak akkor van értelme, ha azt szeretnénk, hogy egy program minél gyorsabban induljon el, ne kelljen várni arra, hogy betöltődjön a memóriába.

A sticky bitet be lehet kapcsolni könyvtárak esetén is. Az ilyen bittel ellátott könyvtárban bárki írhat fájlokat (a többi jognak is rendben kell lennie), de mindenki csak a sajátját törölheti. Ezt a lehetőséget pontosan azért tervezték, hogy az olyan, mindenki által írható könyvtárakban, mint például a /tmp, a felhasználók ne tudják a másik felhasználó által írt fájlokat módosítani, letörölni.

3.1.3.3. Nyomtatás

A Linux nyomtatási filozófiája sok mindenben megegyezik, ugyanakkor sok mindenben el is tér más operációs rendszerek hasonló megoldásaitól. A nyomtatási rendszer

áttekintése tehát nem csak azért szükséges, mert egy felhasználói szinten szintén alaptevékenységként definiálható feladathoz kapcsolódik, hanem azért is, mert segít megérteni a rendszer működési filozófiáját.

A nyomtatás legegyszerűbb módja Linux alatt, ha a kinyomtatni kívánt fájlt közvetlenül a nyomtatónak küldjük. Tekintve, hogy a perifériák a /dev fájlrendszer alatt mind virtuális könyvtárak elérhetőek, egy egyszerű átirányítással a nyomtatás is megoldható.

Nézzük pl. a „*cat level.txt > /dev/lp*” parancsot! A példában a „*cat*” (a paraméterként megadott állomány tartalmát megjelenítő parancs) kimenetét irányítottuk át a /dev/lp szimbolikus linkre, ami tulajdonképpen az aktuális (elsődleges) nyomtató (legyen az lézernyomtató, tintasugaras nyomtató, plotter vagy bármi).

A rendszer biztonság érdekében csak a root (vagy a vele azonos csoportban lévő) felhasználónak van joga közvetlenül a nyomtatóra írni – ezért léteznek azok a programok, amelyek felhasználói jogosultság mellett is elérik a nyomtatót: ilyenek pl. az *lpr, lprm, lpq*.

Az *lpr* parancs a nyomtatás kezdetét felügyeli, majd átadja a vezérlést az *lpd*-nek („line printing daemon” – a démonok rendszerszinten (tehát root jogosultsággal!) futó programok). A nyomtatást ezután ténylegesen az *lpd* végzi. Technikailag amikor az *lpr* végrehajtódik, először átmásolja a nyomtatandó fájlt egy „biztos” könyvtárba (spool könyvtárba), ahol megmarad a nyomtatás befejezéséig. Amint az *lpd*-hez nyomtatási kérés érkezik, elindít magából egy másolatot: ez a másolat fogja nyomtatni a fájlt, amíg az eredeti további parancsokra vár. Mindez lehetővé teszi a párhuzamos feldolgozást, és egyben a párhuzamos feldolgozás valamilyen módon rendezetté tételét.

Az *lpr* parancs sok kapcsolóval rendelkezik (ld. man), aminek a révén a nyomtatási folyamat felhasználói szinten is könnyen felügyelhető. Ezzel együtt egy több-folyamatos környezetben előfordulhat, hogy szükségessé válik a nyomtatási lista megjelenítése is, erre az *lpq* parancs szolgál. Az *lprm* parancs pedig a nyomtatási sorban levő, de még nem nyomtatott kéréseknek a nyomtatási sorból való törlését végzi el.

Magának a nyomtató szolgáltatásnak (*lpd*) kezelésére az *lpc* parancs szolgál. Ez a parancs lényegében egy olyan felügyeleti folyamatot indít el, amely a az *lpd* által kiszolgált nyomtatókra vonatkozó műveleteket határozza meg. Ilyen művelet lehet pl. az egyes nyomtatókhoz való hozzáférés engedélyezése vagy tiltása (természetesen letiltott nyomtatóra nem lehet nyomtatni), a nyomtatók feldolgozási listájának kezelése és a különböző állapot-információk. Az *lpc*-t leginkább olyankor használjuk, amikor több nyomtató tartozik a számítógéphez.

A nyomtatási művelet során alapvető különbségek figyelhetők meg a nyomtatni kívánt állomány tartalma szerint is. Grafikus fájlok esetén a nyomtatás módja általában a nyomtatótól és a nyomtatni kívánt grafikus fájl minőségétől függ.

Mátrixnyomtató esetén nem sok különbség van az egyes nyomtatók között abban, ahogyan kezelik a grafikát, ezért ebben az esetben viszonylag egyszerű dolgunk van. A legjobb eset, ha a nyomtató Epson vagy IBM ProPrinter kompatibilis. Ekkor ugyanis ha a grafikus fájlt átkonvertáljuk postscript-be, akkor viszonylag egyszerűen és megfelelő minőségben nyomtatható.

Lézernyomtató esetén kicsit egyszerűbb a helyzet, ugyanis a legtöbb kompatibilis a PCL szabvánnyal, aminek következtében (ha az alkalmazott program is ismeri ezt a szabványt), a nyomtatás történhet közvetlenül PCL-be. Ha nem, akkor marad a konvertálás. (Ilyen megoldás pl. a NetPBM és Ghostscript programok telepítése, amelyek használatával a grafikus fájlokat közvetlenül nyomtatható formátumba konvertálhatjuk.)

A Postscript fájlok nyomtatásakor hasonló a helyzet: a korszerű nyomtatók nagy többsége rendelkezik (vagy felszerelhető) Postscript interpreterrel, ebben az esetben akár az *lpr*-t használva is nyomtathatunk – a nyomtató felügyel minden tevékenységet (a nyomtatással kapcsolatban).

Ha a nyomtató nem rendelkezik Postscript interpreterrel, akkor marad az előbb említett konvertálás (pl. az előbb említett Ghostscript segítségével).

Az Internetes tartalmak jelentős része a fent említett két formátum mellett ma már jellemzően a pdf formátumot használja, az ilyen fájlok megjelenítésére és nyomtatására (sok más mellett pl.) az Adobe Acrobat Reader-e használható.

Linux-os környezetben említést érdemel még a TeX, mint speciális (nyomtatási) fájlformátum. A legegyszerűbb eset ebben az esetben is a postscript konverzió, azonban ez közvetlen módon nem végezhető el. (A TeX fájlokat előbb DVI fájlokká kell konvertálni – ez a formátum nyomtató-független, és konvertálható ps formátumba.)

Nyomtatók kezelése

A nyomtatási műveletekhez (természetesen) a nyomtató megfelelő beállítása is feltétel. A *Linux* általában képes kezelni a legtöbb korszerű nyomtatót. A nyomtató konfigurálása (az előbb említett lp* parancsokon túl) a grafikus felületről is megtehető: általában a rendszerkonfigurációs beállításokkal kapcsolatos felügyeleti eszközök között (*ld. később*).

3.1.4. Karakteres felület használata

A **shell** (parancsértelmező) a felhasználó és a számítógép közötti kommunikációt biztosítja. A feladatát tekintve hasonló a jól ismert MS-DOS alatti „command.com”-hoz, de sokkal több dologra képes.

A Linux rendszerben több parancsértelmező áll rendelkezésre, de a legelterjedtebb a „**bash shell**”. Az adott rendszerben a rendelkezésre álló parancsértelmezők listáját a „/etc/shells” fájlban tekinthetjük meg.

Minden felhasználó bejelentkezésekor egy parancsértelmező indul el. A parancsértelmező szabványos bemenete és kimenete a terminál. A shell egy promptot jelenít meg (ami egyénileg beállítható), jelezve, hogy készen áll a feladatok végrehajtására. Az aktuálisan futó program egy process-t határoz meg, ebből következően minden futó programhoz a rendszer egy process-t rendel, amely rendelkezik egy azonosítószámmal (*ld. alapfogalmak*)

A processek futása alapvetően kétféle módon történhet: előtérben („foreground”) és háttérben („background”). A két futási mód között az a különbség, hogy a háttérben futó process esetében a shell nem vár a process befejezésére a prompt megjelenítésével, következésképpen egy háttérben elindított process futása közben van lehetőség újabb műveletek elvégzésére. (Megjegyezzük, hogy több folyamat párhuzamos végrehajtására a virtuális terminálok használata lényegesen gyakrabban alkalmazott megoldás.)

A „**bash shell**” szolgáltatásai között fontos megemlíteni, hogy a parancsot nem kell végig begépelni: ha gépelés közben megnyomjuk a „TAB” billentyűt, kiegészíti a parancsot. (Ha még nem egyértelmű a parancs, akkor kétszer egymásután lenyomva a „TAB” billentyűt, megjeleníti a lehetséges parancsokat, amelyek a begépelt karakterekkel kezdődnek.) A „bash shell” tárolja parancsokat, a kiadott parancsok a kurzormozgató billentyűkkel újra megjeleníthetőek, szerkeszthetőek. A shell a működéséhez számos előre definiált és

felhasználó által megadott változót is használ, a rendszer által beállított legfontosabb változók a következők:

- **HOME**: A felhasználó saját könyvtárának elérési útvonala, rövidíthető a '~' szimbólummal.
- **USER**: Felhasználói azonosító.
- **TZ**: A rendszer által használt időzóna.

A változók mellett az egyes shellek a különböző konfigurációs állományokban található beállításokat is figyelembe véve alakítják ki az egyes felhasználók tényleges munkakörnyezetét. A „bash shell" által használt legfontosabb konfigurációs állományok az alábbiak:

- **$HOME/.bash_profile**: inicializációs fájl, a rendszerbe való belépéskor fut le.
- **/etc/profile**: központi inicializációs fájl, a rendszerbe való belépéskor fut le.
- **$HOME/.bashrc**: inicializációs fájl, minden alkalommal futtatásra kerül, ha elindítjuk a shellt.
- **$HOME/.bash_login**: a felhasználó egyéni beállításait tartalmazó állomány, sikeres bejelentkezéskor fut le.
- **$HOME/.bash_logout**: a kilépéskor végrehajtandó tevékenységeket tartalmazó állomány.

3.1.4.1. Alaptevékenységek

Felhasználók bejelentkezése

A több-felhasználós rendszerekben kulcskérdés az egyes felhasználók egymástól való elkülönítése, valamint a felhasználói jogosultságok kezelése. Nézzük most meg, hogy milyen folyamatok mennek végbe, amikor egy felhasználó bejelentkezik a rendszerbe:

Karakteres terminálon a „**getty**" program várja a felhasználók bejelentkezését, amely többek között kiírja a „login" feliratot.

Amikor a „getty" megkapja, hogy a felhasználó bejelentkezik a rendszerbe, akkor meghívja a „login" programot. Ez a program megváltoztatja a felhasználó környezetét, és meghívja a beállított shellt. A „login" program a bejelentkezési kérés jogosságának eldöntésére az „/etc/passwd" fájlt használja. A belépés a rendszerbe csak a megfelelő felhasználónév („usernév") és jelszó („password") megadásával lehetséges. Ennek kiosztása a „root" jogosultsággal rendelkező rendszergazda feladata. A felhasználónak csak a saját jelszó megváltoztatására van jogosultsága.

Például az előbb bemutatott „bash shell" indításakor a következő sorrendben olvassa be a különböző konfigurációs fájlokat, és hajtja végre a bennük szereplő parancsokat:

1. /etc/profile,
2. $HOME/.bash_profile,
3. $HOME/.bash_login,
4. $HOME/.profile.

A fájlok keresése (és végrehajtása) a fenti sorrendben történik, ha valamelyik nem létezik, akkor a rendszer automatikusan a következőt próbálja meg végrehajtani. Természetesen ezek közül a fájlok közül csak azoknak szerkeszthetjük a tartalmát (és ilyen

módon csak azokban állíthatjuk be a saját környezetünket), amelyek a $HOME könyvtárban vannak: „bash shell” esetén tehát a „.bash_profile” fájlban célszerű elhelyezni a kívánt környezeti változókat. Amikor kilépünk a shell-ből, akkor a „$HOME/.bash_logout” fájl hajtódik végre (ha létezik): ebbe a fájlba pl. olyan parancsot szoktak tenni, ami letörli a képernyőt a kilépéskor („clear”).

Bejelentkezés hálózaton keresztül

A hálózaton keresztüli belépésnél mindenki egy fizikai vonalat használ, így itt nem használható az előző leírás. A kapcsolat ilyen esetben úgy jöhet létre, ha két gép valamilyen programja szeretne valamilyen protokollon keresztül kommunikálni egymással. Az „inetd” démon fogadja az felmerült igényeket. Az általunk biztosított szolgáltatást a „/etc/inetd.conf” fájlban határozhatjuk meg. Általánosságban azt mondhatjuk a rendszer alaphelyzetben mindig több szolgáltatást engedélyez, mint amit szeretnénk publikussá tenni. Ezek korrekt beállítása tehát kulcskérdés, mielőtt számítógépünket hálózati eléréssel ruházzuk fel. A szolgáltatások elérését a „/etc/hosts.allow” és a „/etc/hosts.deny” fájlokkal tudjuk szabályozni.

A rendszer leállítása

A rendszer leállítása szigorú szabályok szerint történik. A leállítási folyamatot a „shutdown” program végzi. Minden process-nek tudni kell, hogy most rendszerleállítás jön, tehát le kell zárnia minden nyitott fájlt kilépés előtt. Ha ezt nem tenné meg, a nyitott fájlok sérülhetnek vagy elveszhetnek a leállás során. (Pl. a Linux nem mindent ír azonnal a lemezre (ez egyébként az operációs rendszerek többségénél bevett gyakorlat a háttértárak lassúságának kezelésére, *ld. gyorsítótárak*), hanem a memóriában tárolja a felhasználó által egy mentési műveletben megadott információkat, így ha ezeket végül nem írja ki lemezre, akkor elveszhetnek.) A „shutdown” program megfelelő paraméterezésével érhető el, hogy a rendszer leállítása biztonságosan valósuljon meg (*ld. utasítások*).

A „shutdown” mellett rendelkezésre áll még a „halt” parancs is a rendszer leállítására, és a „reboot” parancs az újraindításhoz.

3.1.4.2. A Linux alapvető parancsai

Habár a Linux rendszerek is rendelkeznek grafikus felülettel, hagyományosan karakteres operációs rendszerről lévén szó tekintsük át röviden a legfontosabb parancsokat. (Az egyszerűség és az áttekinthetőség érdekében a parancsokat ábécé sorrendben soroljuk fel.)

- **bg** PID: a megadott azonosítóval (process ID) rendelkező folyamat háttérbe küldése.
- **cd** [könyvtárnév]: az aktuális könyvtár kiválasztására (beállítására) használható. (Az alapértelmezett aktuális könyvtár az aktuális felhasználó $HOME könyvtára.)
- **chgrp** [kapcsolók] csoportfájlok: a fájlok csoporttagságát módosítja. Ezt a parancsot a rendszer adminisztrátora vagy a megadott fájlok tulajdonosa hajthatja végre.
- **chmod** [mód] név: állomány és könyvtárak hozzáférési engedélyek megadása, elvétele.
- **chown** [kapcsolók] tulajdonos [csoport] fájl: megadott fájlok tulajdonosát és csoportját módosíthatjuk ezzel a paranccsal.
- **cp** [kapcsolók] fájlnév fájlnév: állományok másolása.

- **fg** PID: a megadott azonosítóval rendelkező (háttérben futó) folyamat előtérbe hozása.
- **find** útvonal [opciók]: az útvonalban megadott katalógusból kiindulva, rekurzívan végigjárva a katalógusokat az opcióiban magadott tulajdonságú állományt megtalálja.
- **grep**: egy meghatározott szövegmintát keres egy vagy több fájlban.
- **halt**: a rendszer leállítása
- **job**: a háttérben futó folyamatokra vonatkozó információk megjelenítése
- **kill** PID: futó folyamat (process) befejeztetése, leállítása.
- **ln** régi-fájlnév új-fájlnév: új nevet hoz létre a fájl számára (tulajdonképpen egy hivatkozási név, ún. link („csatolás") jön létre, fizikai másolás nem történik).
- **ls** [kapcsolók] fájl vagy könyvtárnevek: a fájl- és könyvtár-információk megjelenítése.
- **man**: kézikönyv az egyes parancsok szerkezetéről és működéséről („súgó"). A legfontosabb és leggyakrabban használt parancs. Az oldalak a "/usr/man" könyvtárban találhatók vagy a MANPATH környezeti változóban megadott könyvtárban. Kilépés a „man"-ból „q" billentyű lenyomásával történik. Lapozás előre az [Enter] (soronként) vagy a [Space] vagy „f" (oldalanként) billentyűkkel, visszafelé a „b" billentyűvel történhet.
- **mkdir** [kapcsolók] könyvtárnév: könyvtár létrehozása.
- **more**: képernyő oldalanként jeleníti meg a megadott fájlt.
- **mount** [kapcsolók] [eszköz] [csatolási_hely]: új fájlrendszer csatolása egy könyvtárstruktúrába. A meg nem adott paramétereket a parancs az „/etc/fstab" fájl megfelelő bejegyzéseiből veszi. (Fájlrendszer leválasztása: umount.)
- **mv** [kapcsolók] forrás cél: fájlokat vagy könyvtárakat átnevez vagy átmozgat.
- **ps** [kapcsolók]: aktuálisan futó "process"-ek listája
- **pwd**: az aktuális könyvtár teljes elérési útvonalát adja meg.
- **rm** [kapcsolók] fájlok: egy vagy több fájl törlése.
- **shutdown** [kapcsolók] időpont [üzenet]: a rendszer futási szintjét módosítja, vagy lezárja a rendszert. (Azonnali lezáráshoz az időpont értéke: „now".)
- **su** [-] [felhasználó] [argumentum]: a „shell" programot indítja másik felhasználóval. Olyan terminálon való bejelentkezésre használható, amelyet másik felhasználó használ. Ha nem adjuk meg a felhasználót akkor egy root „shell" programot indít. Az új „shell" programot az „exit" paranccsal vagy [Ctrl-D] billentyű-kombinációval zárhatjuk le.
- **tar** [kapcsolók] [archív] [fájlok]: beépített tömörítőprogram, kapcsolói határozzák meg, hogy be- vagy kitömörítést hajtson végre.

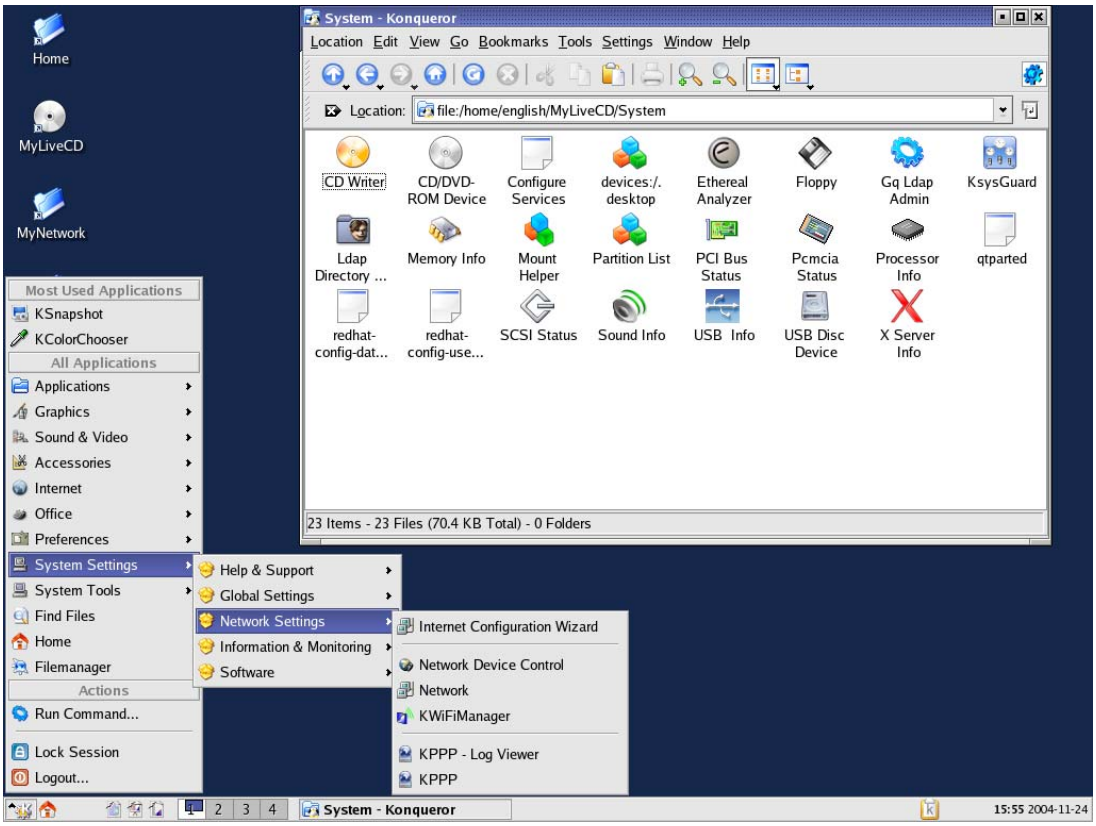
3.1.5. A grafikus felület felhasználói eszközei

3.1.5.1. Az X család

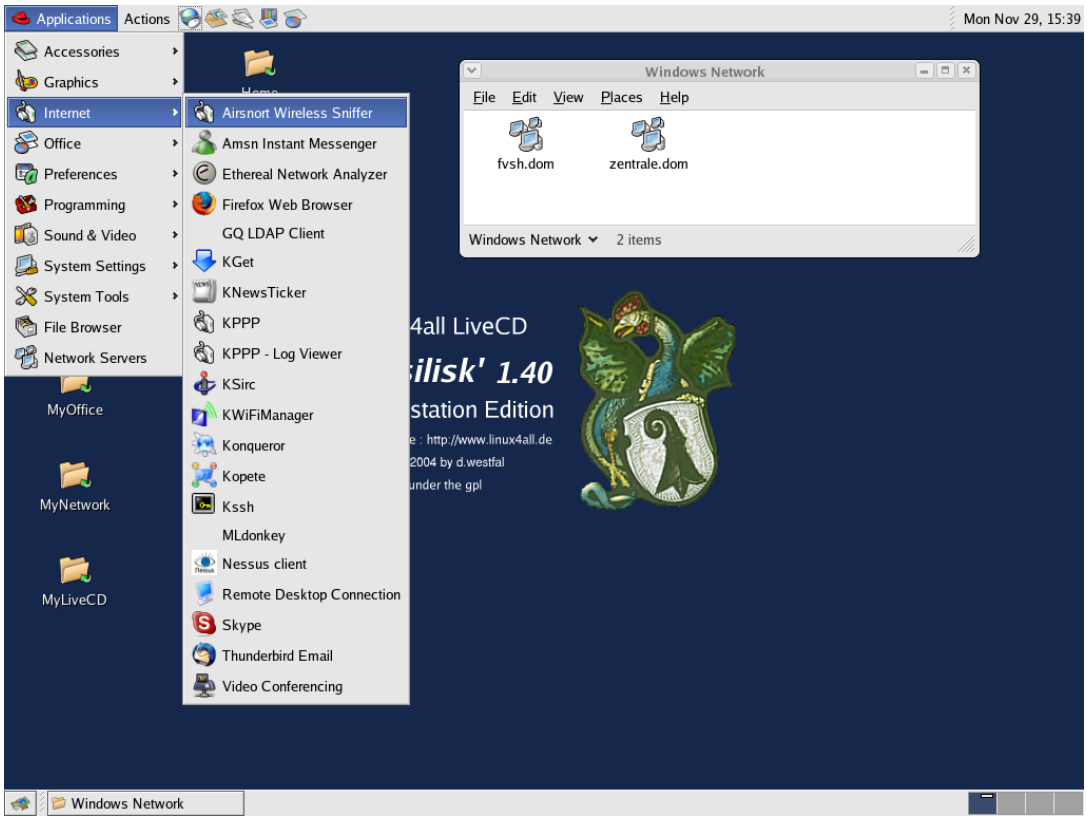
A Linux alapú operációs rendszerek grafikus felhasználói felülete szintén az „XWindow System” architektúrán alapul (*ld. 2.3.1.2. fejezet*). Röviden X-nek is szokás hívni, illetve mivel a szinte mindenhol használt verziója a 11-es, innen az X11 elnevezés. Az XFree86 az „X version 11 Release 6” (röviden X11R6) megvalósítása Intel x86 platformon futó Unix/Linux alapú rendszerekben. Az XFree86 technikailag számos különböző X szervert tartalmaz, amelyek különböző VGA kártyákon futnak. Természetesen az XFree86 is kiegészíthető (illetve kiegészítendő) különböző ablakkezelőkkel („window manager”), amelyek az X felett biztosítják Windows vagy a Mac alatt megszokott felületet és funkciókat. Linux alatt a legelterjedtebb ablakkezelők (pontosabban az X szabvány ablakkezelő definíciójára épülő grafikus felhasználói felületek, „desktop”-ok) a KDE és a GNOME.

A KDE (eredetileg Kool, a későbbiekben egyszerűen K Desktop Environment) alapértelmezett grafikus felhasználói felület pl. a SUSE, a Mandrake vagy a Knoppix disztribúciókban. A KDE projektet Matthias Ettrich indította 1996-ban. Célja az volt, hogy egy komplett grafikus környezetet hozzon létre a Unix/Linux rendszerek számára. Első változata 1998-ban jelent meg, jelenleg a 3.5.x-es verzió érhető el, a 4-es verzió várhatóan 2006 végén jelenik meg.

A GNOME (GNU Network Object Model Environment) fejlesztői (kitalálója Miguel de Icaza) a KDE (általuk vélt) hiányosságait és hibáit kiküszöbölendő alakították ki saját GUI környezetüket, amely a GNU projekt hivatalos grafikus felületévé vált, így alapértelmezett ezekben a disztribúciókban, pl. a Red Hat alatt. Első változata 1997-ben jelent meg, jelenleg a 2.14-es verzió az aktuális, a 3-as változat fejlesztés alatt áll.



3-7. ábra: KDE



3-8. ábra: GNOME

Mindkét felület alatt számos, az alapvető felhasználói igényeknek megfelelő alkalmazást (programot, csomagot) találunk (*3-2 táblázat*): hagyományosnak mondható irodai alkalmazásokat, Internet alapú szolgáltatásokhoz vagy multimédiához kapcsolódó kiszolgáló programokat és néhány egzotikumot is…

3-2. Táblázat: KDE/GNOME programkomponensek

KDE alkalmazások	GNOME programok
Kate: szövegszerkesztő	**gedit:** (egyszerű) szövegszerkesztő
KOffice: irodai alkalmazáscsomag: KWord (szövegszerkesztő), KPresenter (bemutató-készítő), KSpread (táblázatkezelő), Kexi (adatkezelő), stb.	**AbiWord:** szövegszerkesztő **Gnumeric:** táblázatkezelő **The Gimp:** képszerkesztő **Nautilus:** fájl-kezelő
Konqueror: fájlkezelő (és egyben Web-böngésző is!)	**Epiphany:** Web-böngésző (a Firefox kiszorítja)
Quanta Plus: web-szerkesztő	**Evolution:** levelező- és naptáralkalmazás
KMail: levelező rendszer	**Gaim:** üzenetküldő program.
Kopete: azonnali üzenetküldő	**Ekiga:** telefon és VoIP alkalmazás (korábban GnomeMeeting)
Konsole: terminál-emulátor	
amaroK: znelejátszó	**Rhythmbox:** zenelejátszó
K3b: CD/DVD író	**Totem:** médialejátszó
KDevelop: fejlesztői környezet (IDE)	

A két elterjedt GUI felület mellett számos további is létezik, de az egyes megvalósítások között alapvetően csak szemléletbeli különbségek figyelhetők meg, amelyek az általános felhasználói tevékenységet lényegében nem befolyásolják, ezért a rendszerek bemutatását az elterjedt KDE eszközrendszerének ismertetésén keresztül folytatjuk.

3.1.5.2. Bevezetés a KDE-be

A KDE alapfilozófiája szerint a felhasználó számítógépe az Internet egy csomópontja – és mint ilyen, a tartalmához való hozzáférés és a műveletvégzés az Interneten megszokott szabályok és módszereknek megfelelően történhet. Ezért használja a KDE a HTML nyelvet.

Felület: a K menü és a panel

A „K menü” egy egyszerű és hatékony eszköz KDE alatt programok elérésére. A K menüben szereplő programok révén a számítógép használatához szükséges alkalmazások könnyen megtalálhatók és elindíthatók. A K menünek két része van. Az első tartalmazza mindazon alkalmazásokat (tematikus csoportokban: Grafika, Alkalmazások, Internet stb.), amelyek rendelkezésre állnak az adott számítógépen (fel vannak telepítve). A második rész tartalmazza a közvetlen hozzáférést a rendszer alapvető komponenseihez, mint pl. a Kuka, a Beállító központ vagy a Lemez Navigátor. A Beállítás menük közvetlen elérést nyújtanak a rendszer beállításával kapcsolatos műveletekhez (KDE Beállító központ elemeihez), a Navigátor a fájlrendszer elérésével kapcsolatos alapműveleteket egyszerűsíti le, a Panel pedig a képernyő alján található különböző ikonok (ez a panel) vezérlését teszik lehetővé.

A Lemez Navigátor egy eszköz, amely lehetővé teszi a fájl-struktúrához való közvetlen és gyors hozzáférést: könyvtárakhoz, személyes fájlokhoz és a legfontosabb (gyakran használt) alkalmazásokhoz tartozó hivatkozások kezelhetők a segítségével (megfelel az Internet „könyvjelző” fogalmának).

A panel a képernyő alsó sávja. A K menü és a munkaasztalok ikonjain kívül tartalmazhatja más alkalmazások ikonjait is (*3-9. ábra*). Testreszabására a KPanel menü parancsai szolgálnak.

3-9. ábra: A KDE panel

Egérkezelés

A KDE alapértelmezés szerint egyszeres (alapértelmezés szerint az egér bal gombjával történő) kattintással aktiválja a kiválasztott objektumot. Támogatja a környezetfüggő menüket, ezek alapértelmezés szerint az egér jobb gombjával érhetők el, a lenyíló menü tartalma az épp aktuális objektumtól (az egér pillanatnyi pozíciójától) függ.

A harmadik egérgomb (nem három gombos egér esetén a két gomb együttes lenyomásával helyettesíthető) a futó programok (az összes!) listáját jeleníti meg, függetlenül attól, hogy melyik az aktuális munkaasztal.

A KDE is kezeli a „Drag-and-drop” („Fogd-és-vidd”, esetleg „Vonszolás”)-ként ismert módszert, amely szerint egy objektumot kiválasztva és az egérrel elmozgatva a mozgatás befejezésének környezete meghatároz(hat)ja az objektummal végzett műveletet (pl. állományok mozgatásakor, törlés vagy nyomtatás során).

A különböző munkaasztalok

A KDE alapértelmezés szerint négy virtuális grafikus terminállal, ún. „munkaasztallal” dolgozik (ez lehet akár nyolc is). Az egyes munkaasztalok mint önálló X komponensek működnek, egyéni beállításokkal rendelkezhetnek (akár nevesíthetők is).

Segítség

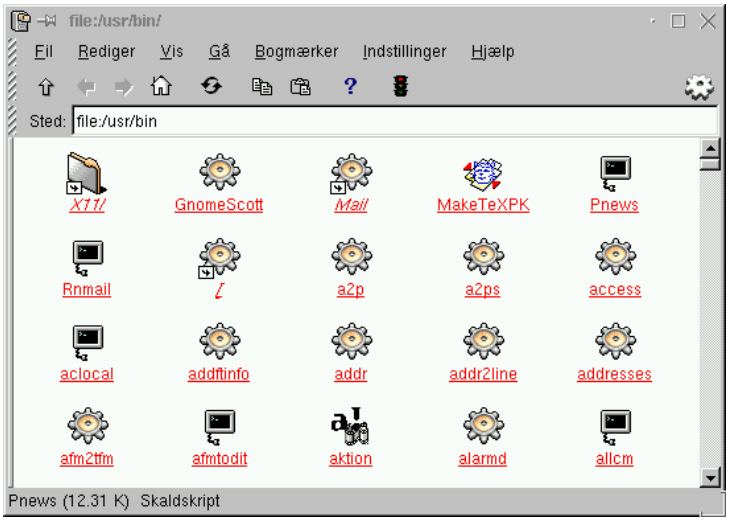
A KDE a rendszerre és kezelésére vonatkozó (on-line) súgóval rendelkezik (természetesen HTML formátumban), alapértelmezés szerint a K menüből érhető el.

3.1.5.3. Felhasználói műveletek

Állománykezelés (KFM)

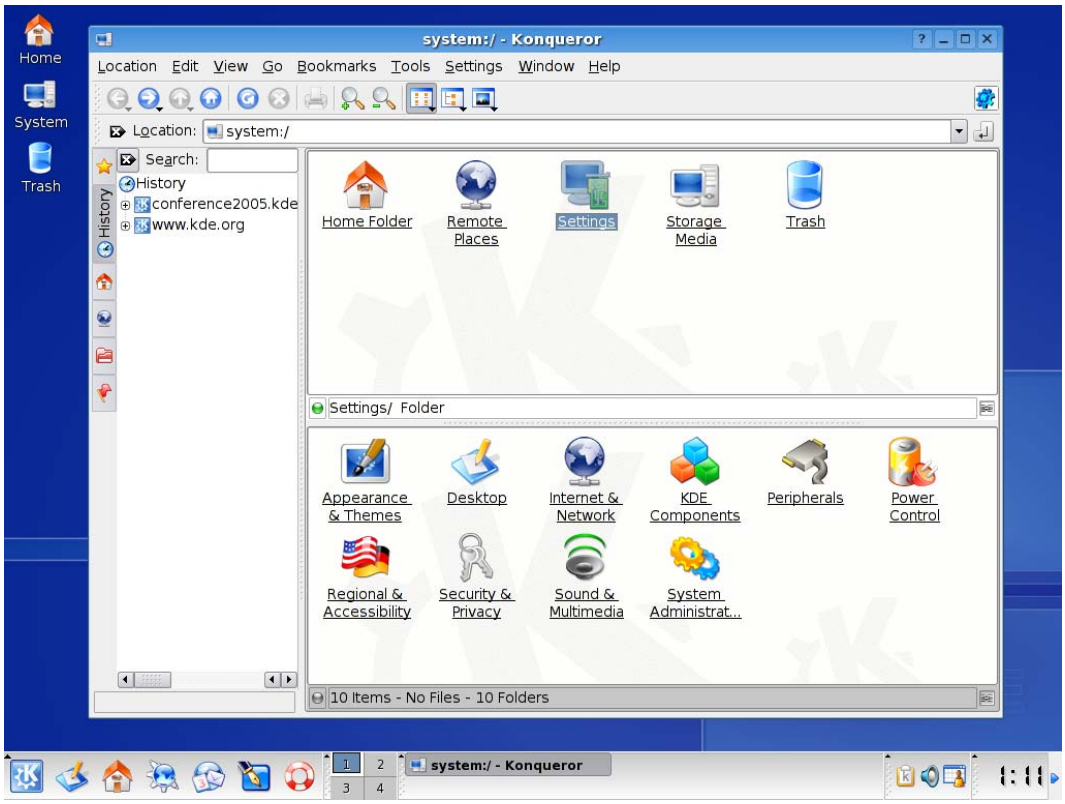
A **KFM** (K File Manager) állomány-kezelő és Internetes böngészőprogram egyben. Gyakorlatilag az összes „nem-alkalmazás” típusú ikonra kattintva a KFM indul el. (A következő generációs változatának neve Konqueror.)

A KFM alapértelmezés szerint egy menü-sorral, egy eszköztárral és egy URL sorral (Címsor) rendelkezik, mindezek alatt jelennek meg az aktuális útvonalnak megfelelő (az URL sor által mutatott) helyen található állományok, könyvtárak, majd pedig a Státusz- (Állapot-) sor következik (*3-10. ábra*). A navigáció (ne feledjük, egyben web-böngésző is!) a kijelölés mellett URL megadásával is történhet:: file:/elérési-útvonal.



3-10. ábra: KFM

A KFM menürendszer többé-kevésbé szabványosnak (illetve szokványosnak) tekinthető: a Fájl és a Szerkesztés menük szerepe a szokásos (létrehozás, keresés, kivágás, beillesztés, ...), a Nézet menüben a megjelenítés módja adható meg, az Előre és a Kedvencek menük pedig a böngészőben szokásos műveleteket tartalmazzák. (előzmények, könyvjelzők).



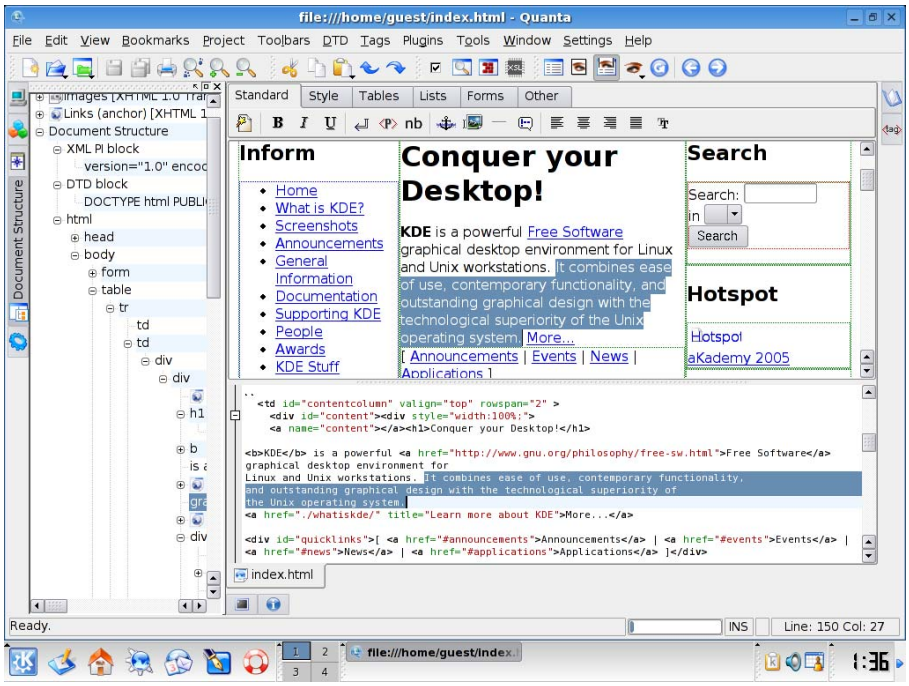
3-11. ábra: KDE a Konqueror-ban

A Nézet menüben az ablak megjelenésével kapcsolatos beállítási lehetőségek mellett megtaláljuk a fájlrendszer sajátosságait tükröző beállítási lehetőségeket is, mint pl. a rejtett fájlok kezelésére vonatkozó beállítások, vagy a fájlrendszer kezdőpontjainak (gyökérkönyvtár: /, személyes könyvtár: $HOME, és a munkaasztal: $HOME/desktop) megjelenítési lehetősége. (Érdekesség: ha a „HTML listázás” opció be van kapcsolva, a KFM a lokális háttértárat is Internetes tartalomszolgáltatóként kezeli, azaz egy könyvtárba belépve annak tartalma helyett az index.html (ha létezik) fájl szerinti tartalom jelenik meg…). A szigorúan vett megjelenítésre vonatkozó beállítások a jobb oldali munkaablak kinézetére vonatkoznak, azaz azt határozzák meg, hogy az állományok megjelenítése „Nagy ikonok”, „Kis konok”, „Szöveges” vagy „Részletes” formájában történjen-e.

A hagyományos értelembe vett állománykezelési feladatokkal kapcsolatban a KFM semmi újdonságot nem tartalmaz: a műveletek elvégezhetők a menüsor megfelelő parancsainak vagy a környezetfüggő menük parancsainak a segítségével, konfigurálására a Beállítások menü szolgál.

A Beállítások között említést érdemel (elsődlegesen gyakorlati alkalmazhatósága miatt) a „Fa nézet kövesse a navigációt” opció: ezzel azt határozhatjuk meg, hogy a bal és a jobb panel elmozdulása szinkronizálódjon.

A KFM a fájlok kezelésével is az Interneten használatos szabványokhoz igazodik, mint amilyen a MIME is. A MIME alapgondolata a fájltípusok alkalmazásokhoz való társítása (a fájltípusokat alapvetően a névben szereplő azonosító rész határozza meg: ez megfeleltethető más operációs rendszerek „fájltípus” fogalmának, de a Linux-ban nincsenek típusok!). Vannak előre definiált típus-alkalmazás összerendelések, illetve van lehetőség ezek megváltoztatására és újabbak létrehozására is.

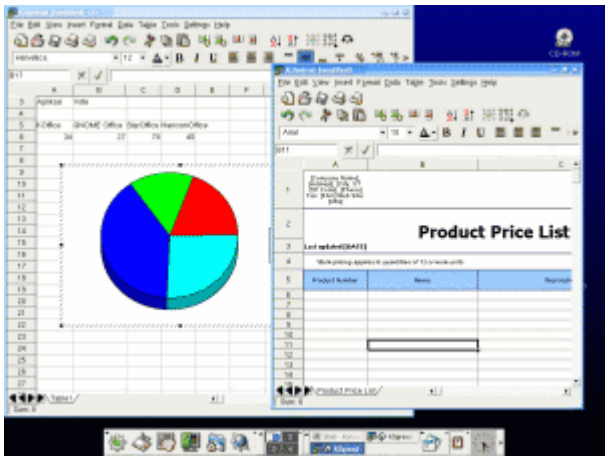


3-12. ábra: Conquer your Desktop! – Hódítsd meg a (munka)asztalt!

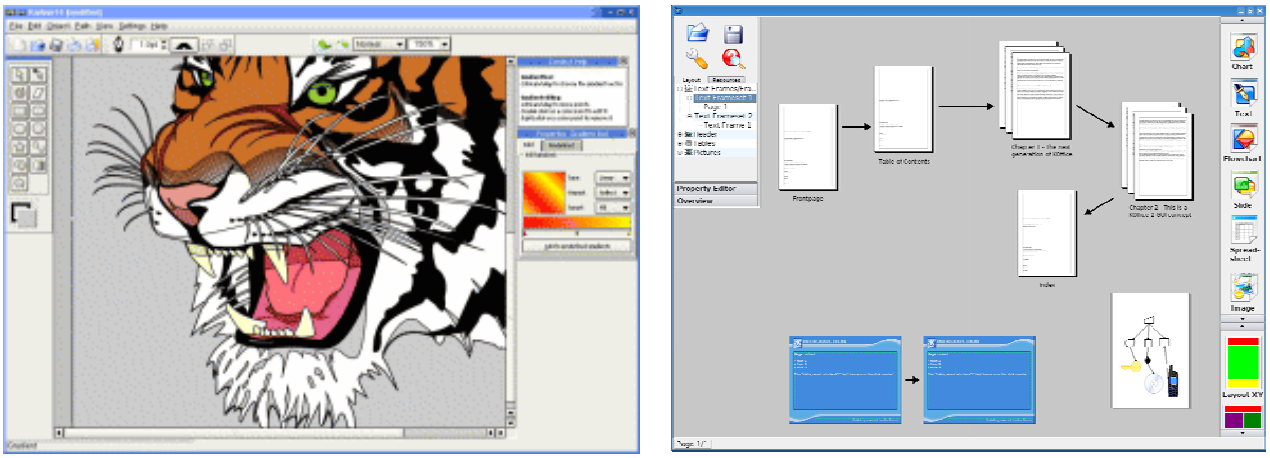
A KFM egy Internet böngészőprogram is, támogatja a HTML nyelv legutolsó változatát és a jelentősebb Internetes szabványokat és alkalmazásokat (pl. a JavaScript-et, stb.). A böngészőként történő alkalmazás során az Internet-kapcsolat KFM által használt beállításait (paramétereit) szintén a Beállítások menü alatt adhatjuk meg, illetve módosíthatjuk..

Alkalmazások[17]

A KDE teljes értékű irodai szoftvercsomaggal rendelkezik (KOffice, *3-12. ábra*), amelyben a hagyományos irodai munkavégzéshez szükséges valamennyi alapfeladat elvégzéséhez található alkalmazás.



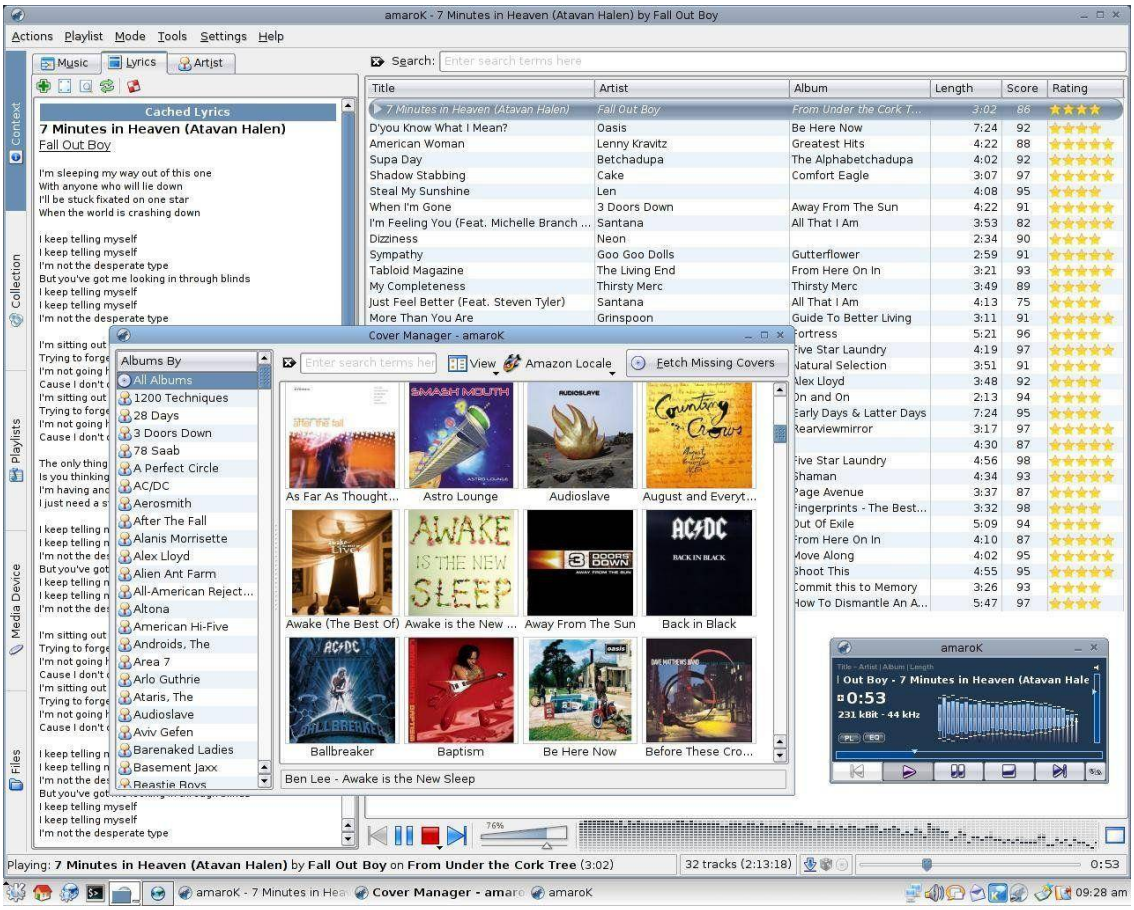
[17] http://www.kde-apps.org/



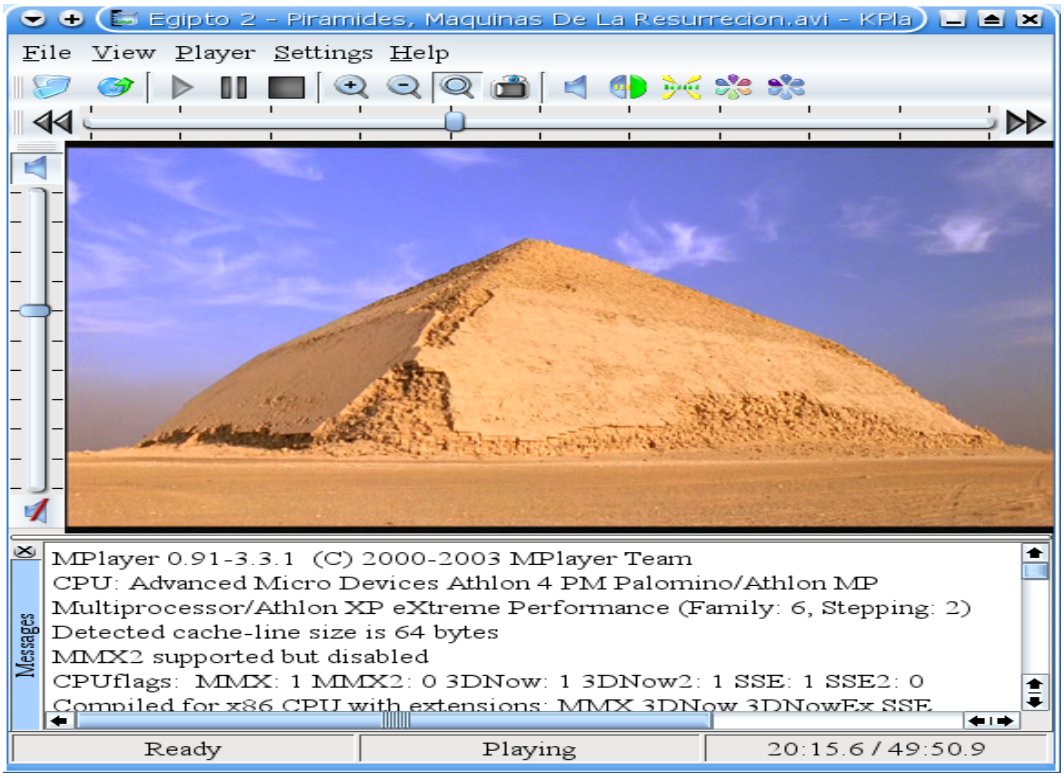
3-13. ábra: KOffice komponensek

Az egyszerűbb felhasználói tevékenységek támogatására is számos alkalmazás áll rendelkezésre KDE-ben, pl. beépített szövegszerkesztőből több is: kedit, kwrite, kate. A kedit egy kis méretű, egyszerű szövegszerkesztő („text editor”), formázatlan szövegek kezelésére alkalmas (pl. a konfigurációs fájlok szerkesztésére). A kwrite és a kate két hasonló elvű program, a kedit-hez képest nagyobb funkcionalitással rendelkeznek, pl. (a KWord-höz képest korlátozott) formázási műveletek támogatásával. Érdekessége a dokumentum típusának függvényében történő belső kezelés képessége (pl. a megjegyzéseket eltérő színnel/formátummal jelöli, emiatt elsősorban pl. nyelvi fejlesztőeszközként használható).

A multimédia-támogatással kapcsolatban megtalálható a csomagban az egyszerű képszerkesztő (kPaint), ikonszerkesztő, képnézegető és képlopó (Ksnapshot) programok, audió- és videó-szerkesztők és lejátszók (amaroK: *3-13. ábra*, KMedia, KPlayer: *3-14. ábra*, vivia, xmms).



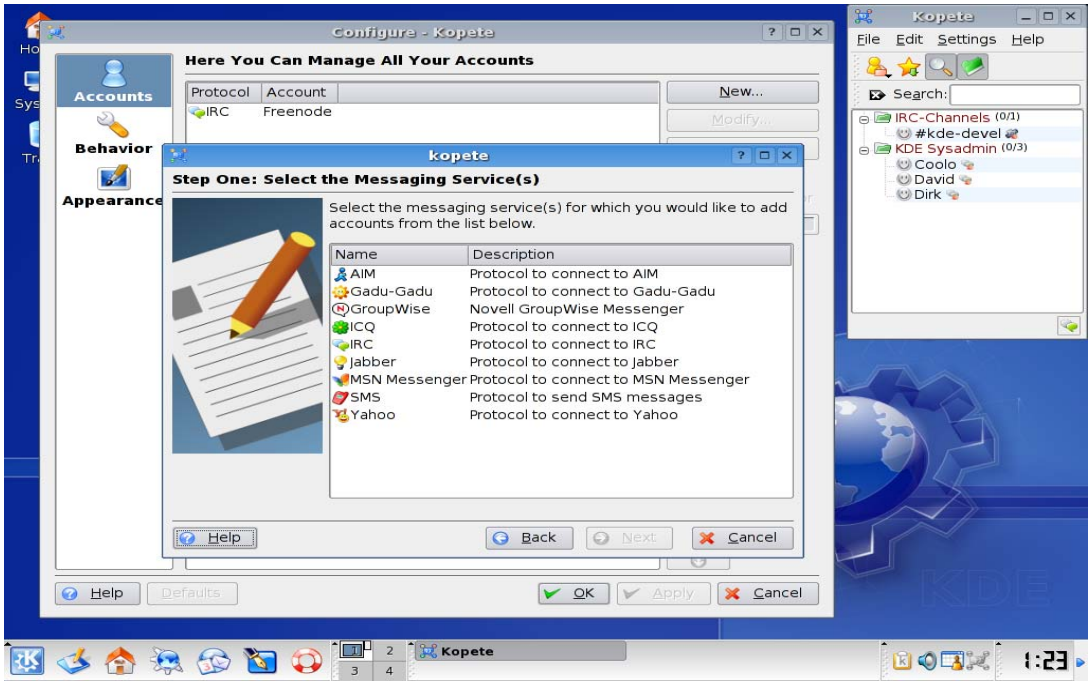
3-14. ábra: amarocK: a mádia-központ



3-15. ábra: KPlayer: a videó-lejátszó

A Linux az Interneten készült, csakúgy, mint a KDE – természetes hát, hogy szinte az összes szóba jöhető hálózati szolgáltatással kapcsolatban rendelkezik valamilyen

alkalmazással. Böngésző programja alapvetően a Netscape vagy a Mozilla (ne feledjük: a Konqueror egyben böngésző is!), levelezésre a Kmail szolgál, de teljes körű Internet támogatást nyújt: kopete (chat), kftp (fájl-átvitel), Kommute (állomány-megosztás), KnetworkManager (hálózat-felügyelet) (pl. *3-15., 3-16. ábrák*).



3-16. ábra: Üzenetküldő rendszerek a KDE alatt

3-17. ábra: WebCam: élő, on-line beszélgetés Linux alatt

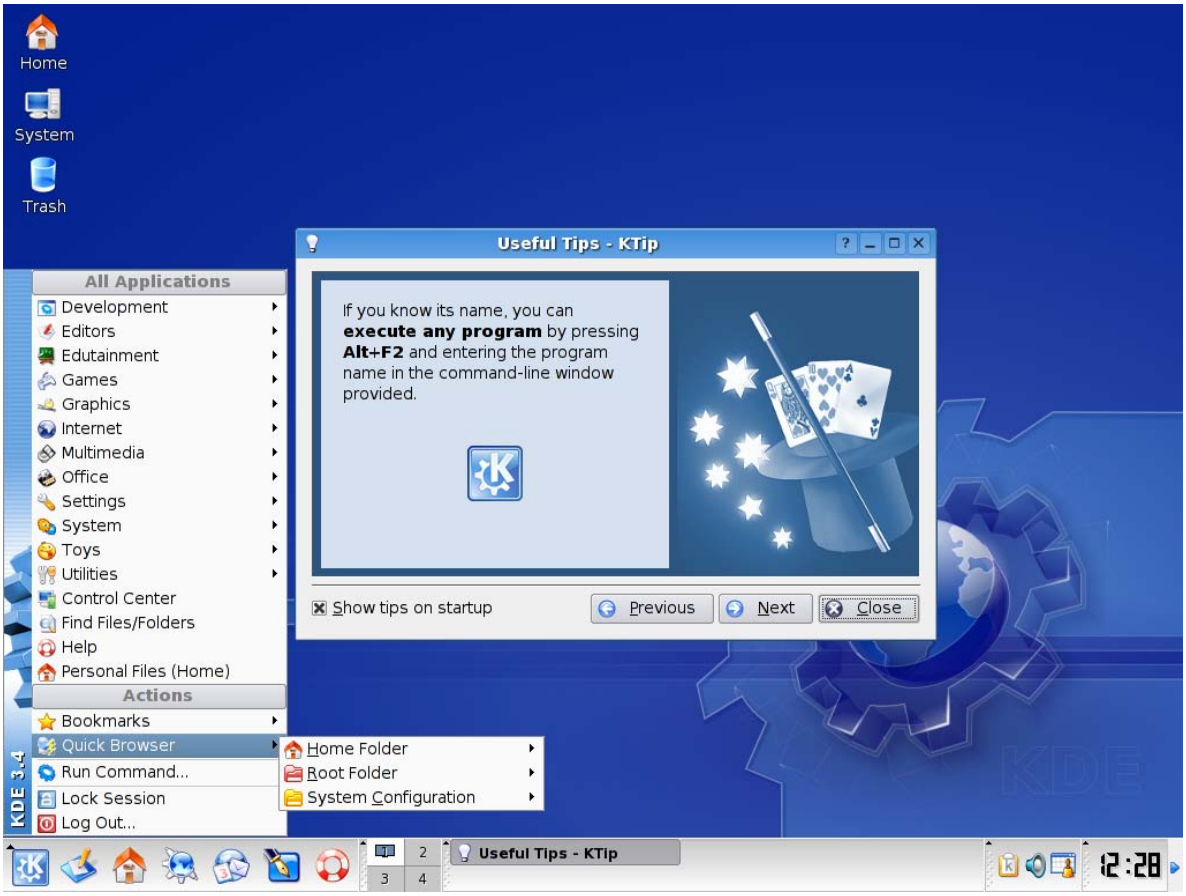
A teljesség igénye nélkül az összes alkalmazás felsorolásának mellőzésével két megjegyzés a grafikus felhasználói felület használatával kapcsolatban:

- ahogy azt a karakteres felület kezelésének bemutatásakor láttuk, a felhasználói bejelentkezéskor végrehajtandó tevékenységek megadására a $HOME/.bash_profile fájl szolgál. Grafikus felületen hasonló szerepet tölt be a StartUp könyvtár: azok a programok, amelyek ebben a könyvtárban helyezkednek el, automatikusan elindulnak (megjegyezzük, hogy a gyakorlatban maguk a programok átmásolása-áthelyezése helyett csak hivatkozásokat (link) szokás ebbe a könyvtárba elhelyezni!).
- a másik érdekesség a cserélhető háttértárak kezelésével kapcsolatos: a karakteres felületen megszokott fel- illetve lecsatolásra (mount, umount) a leggyakrabban használt cserélhető perifériák esetében (floppy és CD) nincs szükség: a munkaasztal alapértelmezés szerint tartalmaz két hivatkozást ezen műveletek elvégzésére. (A perifériák egy kattintással felcsatolhatóak (kis zöld kocka jelenik meg az ikonon), a helyi menüből (jobb gomb) pedig lecsatolhatóak.)

3.1.5.4. KDE Beállításai

A munkaasztal testreszabása

A KDE Beállító Központ (Control Center. *3-17. ábra*) ahonnan kiindulva beállítható a grafikus felület. A beállító központban nyolc rész található, amelyek téma szerint csoportosítják a beállításokat. A felhasználói szempontból legfontosabbak (leggyakrabban használtak) a következők:



3-18. ábra: A K Menü szerkezete

Information: a számítógéppel és az operációs rendszerrel kapcsolatos információk (megszakítási rendszer, memória, DMA-csatornák, az XServer beállításai, stb.)

KFM: a KFM beállításai

Look & Feel: a munkaasztal testreszabása. A Háttér részben a virtuális munkaasztalok megjelenési módja adható meg. Tapéta nélküli munkaasztal esetén megadható a háttérszín (akár két szín is), a Tallóz gomb segítségével kép helyezhető el a háttérben (alapértelmezés szerint JPEG formátumú). A „Panelba dokkolás" opció segítségével a KDE megjelenít egy ikont a panel jobb oldalán, amivel ez a beállítási ablak közvetlenül elérhető (hasznos lehet, ha valaki gyakran módosítja a hátterének beállítását…).

A Szegélyek részben az aktív munkaasztal szélének viselkedése határozható meg: az egér segítségével közvetlenül mozoghatunk a munkaablakok között, (Aktív ablak szélek), illetve az ablakok automatikusan a munkaterület széléhez igazíthatók (Varázs szélek).

A Színek rész a munkaasztal egyes elemeinek megjelenését, a Betűtípusok pedig a használt karakterkészletek megadására, módosítására szolgál. Az Ikonok az ikonok automatikus elrendezésének pontosságáról, a Témakezelő előre definiált (a KDE-vel kapott vagy az Internetről letöltött) beállítások használatáról rendelkezik. (A Stílus részben pl. a Windows vagy a Mac OS X felületének megfelelő kinézet is beállítható…)

A Képernyővédő természetesen a képernyővédő beállítására szolgál. A védelem ellátható jelszóval, ekkor a képernyővédő kikapcsolásához meg kell adni az aktuálisan bejelentkezett felhasználó jelszavát.

Window Behavior & Decoration: ebben a menüben az ablakok megjelenésével és működésével kapcsolatos beállítások tehetők. A menü egyes elemeinek használata a rendszer működésével kapcsolatos mélyebb ismeretek meglétét is feltételezi, de bizonyos beállítások ezek megléte nélkül is elvégezhetőek. Ilyen pl. a gombok helyzetének meghatározása az ablakok fejlécében, az egymás után megnyitott ablakok átfedési beállításai vagy az egérnek az ablakok kezelésére vonatkozó műveletei.

A fókusz azt határozza meg, hogy mi történjen egy inaktív ablakkal, ha az egér rámutat. Négy lehetőség van:

- Fókusz rákattintással: az ablak kattintásra megkapja a fókuszt, és előtérbe jön.
- A fókusz az egeret követi: mindig az az aktív ablak, amelyikben az egér mutatója található, de az aktív ablak takarásban marad, nem jön a felszínre.
- Klasszikus „fókusz követi az egeret": ugyanaz, mint az előző, csak az [Alt+Tab] billentyű-kombinációval is lehet az ablakok között váltani.
- Klasszikus „laza fókusz": mint az előző, de ha a kurzor elhagy egy ablakot, az addig aktív marad, amíg a kurzor egy másik ablakba át nem lép.

Sound: a hanggal és a hang manipulálásával kapcsolatos programok és eszközök beállítása.

Network: hálózati beállítások.

Peripherals: az alapvető beviteli eszközök (egér, billentyűzet) jellemzőinek beállítása.

Personalisation: személyes beállítások.

Power Control: energiatakarékossági beállítások.

System: rendszer karbantartására szolgáló beállítások.

3.1.6. Telepítés, karbantartás, felügyelet

3.1.6.1. Telepítés

A Linux rendszer telepítése lehet komoly szaktudást igénylő informatikai kihívás és egy egyszerű ujj-gyakorlat egyaránt. A kettő között alapvetően az dönt, hogy az általunk

használni (telepíteni) kívánt disztribúció milyen mértékben támogatja magának a telepítésnek a folyamatát. Az egyes disztribúciók között számos eltérés figyelhető meg, de általánosságban a következőkben bemutatásra kerülő telepítési séma (ami a Red Hat telepítésének vázlatát mutatja be) a legtöbb rendszer esetében alkalmazható (vagy legalábbis támpontként szolgálhat). Reméljük, hogy a leírás végére az Olvasó is képes lesz önállóan, a saját gépén egy Linux rendszer feltelepítésére.

A szükséges hardver

A Red Hat esetében a minimális hardverkiépítés: Intel486, 8 Mbyte RAM, 100 Mbyte szabad lemezhely, monokróm (VGA) videokártya, monitor. (A Linux egyébként működik akár egy 386/4 MB RAM/20 MB HDD paraméterekkel rendelkező gépen is, de ez valóban csak az alaprendszer…) A minimum itt tényleg a szükséges legkevesebb: a fenti paraméterekkel rendelkező számítógépre grafikus felületet már nem telepíthetünk... A Red Hat esetében az optimális kiépítés (szerencsére a mai számítógép-konfigurációkat figyelembe véve már ennél is csak „erősebb” gépekkel találkozhatunk): Intel-P3/AMD-K5 vagy ezeknél jobb processzor, 16 vagy több MB RAM, 500 MB szabad lemezhely, SVGA videokártya és monitor.

A telepítés előkészítése

A Linux rendszer sokféleképpen telepíthető: telepítő CD-ről, másik merevlemezen elhelyezkedő telepítési képből („image fájl”-ból), hálózatról (pl. FTP szerverről). Mivel az utóbbi két eset lényegében az elsőnek a telepítőkészlet helyében eltérő speciális változata, az általános telepítés menetét a CD-ROM-ról történő telepítésen keresztül mutatjuk be.

A telepítő program elindításához kell egy üres 1,44-es floppy, ez lesz a rendszerindító lemez („boot disk”). (A telepítés indítható közvetlenül a CD-ről is – pl. DOS alól – a \REDHAT\DOSUTILS\AUTOBOOT.BAT paranccsal, amennyiben a gépen elérhető más operációs rendszer.) A boot lemez DOS alól (vagy olyan rendszerekből, ahol DOS parancssor elérhető) a *RAWRITE.EXE* programmal, Unix/Linux rendszerek alól a *dd* paranccsal készíthető el. (Mindkét esetben két paramétert kell megadni, elsőként a felírandó image-file nevét (általában boot.img), másodikként a lemezt tartalmazó meghajtó azonosítóját.)

A Linux számára két önálló partícióra van szükség. Az egyikre kerül maga a rendszer, a másik lesz a „swap” partíció, ahol a rendszer a virtuális memóriát „tárolja”. Ha nincs elég nagy particionálatlan terület a merevlemezünkön, egy meglévő partíciót kell törölni, vagy a méretét csökkenteni. (Különleges programokkal bizonyos partíciók adatvesztés nélkül átméretezhetők, ilyen pl. a Red Hat esetében a telepítő CD-n megtalálható *FIPS.EXE*, vagy a Partition Magic nevű – kereskedelmi – program.)

A boot lemez elkészítése és a szükséges hely megteremtése után állítsuk le a számítógépet, majd indítsuk újra a boot lemezről!

A telepítés kezdete

A Linux telepítő indulásakor az operációs rendszer detektálja és felismeri az alapvető hardverelemeket (ez bizonyos esetekben sokáig is eltarthat – ne legyünk türelmetlenek!), a megjelenő kérdésekre értelemszerűen kell válaszolni. (A menükben a kurzormozgató és a [TAB] billentyűkkel lehet mozogni, kijelölni a szóközzel, kiválasztani az [Enter]-rel, az [F12] egyenértékű az *OK* parancsgomb lenyomásával.) Ezután következik a legtöbb nem szokványos hardverelem felismerése. (Ha olyan eszköz is van a gépben, amelynek felismerésének a boot-oláskor kellene megtörténnie, *boot:* promptnál még az induláskor paraméterként kell megadni, pl.: *boot: linux hdc=cdrom*).

A CD-ről történő telepítéshez válasszuk a CD-ROM-ot mint forrást, majd adjuk meg annak típusát. (Manapság a CD-ROM-meghajtók nagy többsége IDE vagy SCSI vezérlővel működik.)

Amennyiben SCSI eszköz is van (vagy csak az van) a számítógépben, először be kell tölteni a megfelelő SCSI meghajtó programokat. (Ha a felsorolt SCSI meghajtók egyike sem tudja kezelni a hardvert, akkor sajnos egy túl speciális eszközzel rendelkezünk. Ebben az esetben megkísérelhetünk egy újabb (vagy másik) Linux verzió után nézni, vagy kicserélhetjük a vezérlőt egy ismertebb típusra.)

Particionálás

Mint azt már az elején említettük, a telepítéshez legalább két Linux partícióra van szükség: egy Linux native (82), és egy Linux swap (83) partícióra. Ezek bármelyike lehet akár elsődleges („primary”), akár logikai meghajtó (egy kiterjesztett („extended”) partíció részeként). (A DOS-szal ellentétben a Linux akár több – maximum négy – elsődleges partíciót tud egy merevlemezen kezelni, de a kiterjesztett partícióval együtt is legfeljebb négy partíciónk lehet. Ez a határ természetesen nem vonatkozik az kiterjesztett partíciókon belül létrehozott logikai meghajtókra.)

Az, hogy két partícióra van szükség, nem jelenti azt, hogy ne használhatnánk többet is: ha van elég hely, előnyösebb egynél több natív Linux partíció használata. Érdemes külön partícióra tenni a programokat (/usr), a leveleket és más gyakran változó dolgokat (/var), a felhasználók adatait (/home) és az ideiglenes tárolásra használt területet (/tmp).

A fent felvázolt sémában a rendszer gyökérkönyvtára (/) számára elég 5-10 MB hely, a /usr mérete nehezen becsülhető (általában néhány száz MB és 1-2 GB között), a levelezés mértékétől függően 20-100 MB kell a /var alatt, a felhasználók számától függően 10-50 MB a /tmp alatt, és a tárolt anyagok mennyiségének, a felhasználók számának függvényében méretezzük a /home-ot. A /swap területet általában a fizikai RAM kétszeres kell, hogy legyen.

A tényleges particionálást a Linux *fdisk* programjával végezhetjük el. Ez az egyszerű szöveges program nagyon hatékony, a legfontosabb kapcsolói (tömören összefoglalva):

- *n* - új partíció létrehozása
- *p* - meglévő partíciók listázása
- *t* - meglévő partíció típusának megváltoztatása
- *d* - partíció törlése
- *q* - kilépés mentés nélkül
- *w* - partíciós tábla felírása a merevlemezre, majd kilépés

Az fdisk alapértelmezés szerint Linux (natív) partíciókat hoz létre, ennek kódja 82. A /swap partíció létrehozásához előbb egy megfelelő méretű Linux partíciót kell létrehoznunk, majd annak típusát 83-ra (Linux swap) állítani.

A particionálás után a rendszer figyelmeztet, hogy érdemes lehet újraindítani a gépet – erre azonban általában *nincs szükség*! (Az új partíciós táblát a kernel beolvassa és tudomásul veszi, ha mégsem, akkor kell csak újraindítani a gépet, majd a particionálást kihagyva folytatni a telepítést.) A particionálás után a swap partíciókat inicializálni kell!

Ezek után jön a „root”-partíció kiválasztása, ahova a „root” fájlrendszer (/) kerül. A partíciót kiválasztása után formázhatjuk – ez elengedhetetlen, kivéve, ha egy már formázott Linux partícióról van szó.

Végezetül egy megjegyzés: a telepítés során csak azokat a partíciókat adjuk hozzá a rendszerhez, amelyek a telepítéshez szükségesek. (Ennek egyrészt a partícionálás és formázás

időigénye, másrészt a telepítő fdisk programjának a grafikus eszközök (pl. linuxconf) használatához képesti nehézkessége az oka…)

A csomagok kiválogatása

Ha a particionálással és a formázással készen vagyunk, jöhet a tényleges telepítés: ehhez ki kell választanunk a telepítendő komponenseket. A Red Hat telepítője nem ajánl fel előre megtervezett konfigurációt, legalább nagy vonalakban nekünk kell eldönteni, mire szeretnénk használni a gépet.

A csomagok kiválasztása kétszintű művelet, melyből a második kihagyható. Az első szinten csoportokat választhatunk ki, azaz eldönthetjük, hogy feltesszük-e pl. a fejlesztő eszközöket, a hálózati komponenseket vagy a játékokat, stb., vagy kérhetjük az összes program telepítését is („Everything"). *Vigyázat: a teljes rendszer helyigénye több, mint 500 Mbyte, és általában nincs szükség mindenre!* Ha itt kiválasztjuk a „*Select individual packages"* opciót, akkor a második szintre jutunk, ahol a programcsomagokat egyenként jelölhetjük ki. Itt is csoportokba van rendezve a telepíthető programok listája, de ha kiválasztunk egy csoportot, áttekinthetjük a csoportba tartozó csomagokat (egy-egy csomagról rövid leírást kaphatunk az [F1] gomb lenyomásával).

Vannak olyan csomagok, amelyek elengedhetetlenek a rendszer működéséhez (ezeket nem lehet kihagyni a telepítéskor), továbbá az egyes csomagok között fennállhatnak függőségek („dependencies"), azaz egy adott program működéséhez melyik másikat kell még telepíteni. Ha kihagyunk egy csomagot, amire egy másiknak szüksége van, a telepítő automatikusan felajánlja a kihagyott csomag telepítését.

Ha sikeresen kiválasztottunk mindent, a függőségek is rendben vannak, elindul a telepítés. Először az alaprendszeré, majd sorban a kiválasztott csomagoké. A telepítés közbeni üzeneteket később a /tmp/install.log fájlban megtalálhatjuk. A telepítés a számítógép kiépítettsége, a CD-ROM (hálózati telepítés esetén a hálózat) sebességétől és a kiválasztott csomagok mennyiségétől függően tíz perctől másfél óráig is tarthat.

Alapvető beállítások

A programok telepítése után, mielőtt újraindítva a gépet elkezdhetnénk használni új rendszerünket, pár alapvető beállítást el kell végeznünk.

Először a megfelelő billentyűzet kiválasztását a megjelenő menüből, ahol találunk magyar kiosztást is (később a */usr/sbin/kbdconfig* paranccsal változtathatunk ezen), majd az egér beállítása következik. Érdemes tudni, hogy a Linux rendszert háromgombos egérre tervezték, azaz az egér mindhárom gombjára szükségünk lesz a kezelésekor. Kétgombos egérnél emulálható (*Emulate 3 Buttons*) a harmadik gomb mindkét gomb együttes lenyomásával. Ha az egér nem a soros porthoz kapcsolódik (például PS/2 mouse porthoz), típusától függetlenül a megfelelő porthoz tartozó típust kell kiválasztani. Az egérbeállítást a későbbiekben a */usr/sbin/mouseconfig* paranccsal módosíthatjuk.

Ezután - ha telepítettük - a grafikus felület (XWindow System) beállítása következik. Ennek beállítását az *Xconfigurator* program végzi. A beállító program először a videokártyák listáját mutatja, ahonnan a sajátunknak megfelelő típust kell kiválasztanunk. Ha ezzel vagy a videovezérlő más adatával nem vagyunk tisztában, egy másik virtuális konzolon a *SuperProbe* futtatásával minden adatot megtudhatunk róla, de csak a már működő rendszerből, azaz ez esetben itt át kell ugrani a grafikus felület beállítását. (Vigyázat! A SuperProbe program a rendszer teljes lefagyását okozhatja, mivel közvetlenül a videokártyát használja!) A grafikus felület beállításának elhalasztásából semmiféle problémánk nem adódhat, azt bármikor megtehetjük az említett Xconfigurator vagy az *XF86Setup* programmal.

Ha a videokártyánk típusa nem található a listában, több lehetőségünk is maradt. A legegyszerűbb elhalasztani a beállítást, upgrade-elni a grafikus rendszert. Másik lehetőség: keresünk egy a kártyánkkal kompatibilis vezérlőt, és azt használjuk.

Ha sikerült kiválasztani a megfelelő vezérlőt, a telepítő felteszi a hozzá szükséges drivert, majd jöhet a monitor kiválasztása. Ismét bőséges listát kapunk, sokféle monitorral. Ha a monitorunk megegyezik a listán szereplők valamelyikével, szerencsénk van. Általában a miénkhez nagyon hasonló másik monitort kell kiválasztani. Végső esetben a *Generic monitor*, vagy ha jobb monitorunk van, akkor a *Generic multisyncron* monitor kiválasztása vezethet eredményre, illetve ha a monitor kézikönyvében megtaláljuk a monitor adatait, akkor a *Custom* kiválasztásával azokat megadva pontosan be tudjuk állítani az adatokat. Figyelem! Vigyázzunk, hogy a kiválasztott monitor semmiképpen se legyen jobb, mint a mienk, mivel ezzel túlvezérelhetjük a monitorunkat, és ez akár tönkre is teheti.

A monitor kiválasztása után meg kell adnunk a videomemória mennyiségét is. Később ezt az adatot is megtudhatjuk a SuperProbe futtatásával, ha a dokumentációból nem derül ki. Kevesebb RAM-ot megadva a grafikus felület működni fog, de nem tudja maradéktalanul kihasználni a videokártya képességeit. Több memória megadása nem okozhat fizikai károsodást, de előfordulhat, hogy a grafikus rendszer nem jól indul el.

Végül a *ClockChip* kiválasztása következik. Ha biztosan tudjuk, hogy van, és tudjuk, hogy milyen órajel-generátor van a videokártyánkon, azt válasszuk ki. Ellenkező esetben a legbiztosabb a *No ClockChip* opció kiválasztása. Ezután beállíthatjuk az elérhető felbontások és színmélységek közül a számunkra leginkább megfelelőket.

Érdemes tudni, hogy felbontást menet közben a [Ctrl] [Alt] [szürke +], illetve [szürke -] gombkombinációkkal válthatunk, a színmélységet viszont csak a grafikus felület újraindításával módosíthatjuk.

A már futó rendszerünkben a grafikus felületet belépés után a *startx* paranccsal indíthatjuk el. Ilyenkor mindig a 8 bites (256 színű) mód indul el.

A telepíthető komponensek

Ezután, ha nem hálózatról telepítettük a rendszert, lehetőségünk van a hálózat beállítására vagy újrakonfigurálására. Erre csak akkor van szükségünk, ha a gépünkben van valamilyen hálózati kártya, és azt használni is szeretnénk (ez utóbbi javasolt). A PPP vagy más modemes hálózati beállítást nem itt kell elvégezni. A kérdésekre értelemszerűen válaszoljunk. Alapvető beállításokhoz szükséges adatokért keressük meg a cég vagy intézmény hálózati szakembereit, otthoni gép esetén a dokumentáció tanulmányozása mellett próbálkozzunk. Fizikai kárt nem okozhatunk vele.

A gép órájának beállítása eltérő lehet attól függően, hogy használunk-e másik rendszert is a gépen. Linux alatt az az ideális, ha a gép (CMOS) órája GMT szerint jár, és a helyi időt a Linux rendszer kalkulálja. Ez lehetővé teszi a téli/nyári időszámítás közötti automatikus átállást. Ha viszont DOS-t vagy DOS alapú rendszert is használunk, akkor az a helyi időt várja a CMOS órától, így azt kell használnunk. Ennek megfelelően értelemszerűen állítsuk be az időt. Az időzóna-listában szerepel Budapest is, Europe/Budapest címszó alatt. A */usr/sbin/timeconfig* paranccsal bármikor változtathatunk a beállításon.

Az idő beállítása után egy jelszót kell adnunk a *root user* számára. A rendszeren telepítés után a root az egyetlen aktív felhasználó (van sok más, szerverekhez, szolgáltatásokhoz kapcsolódó, elengedhetetlen, de ember által sosem használt azonosító is). Ő a rendszergazda. A root usernek mindenhez van joga, és bármit megtehet, ezért ezen a néven dolgozni veszélyes és célszerűtlen. A későbbiekben leírjuk, hogyan lehet létrehozni új

felhasználói azonosítót. A root jelszavának legalább nyolcbetűsnek kell lennie, és javasolt a kis-, nagybetűk valamint számok használata a jelszóban a nagyobb biztonság érdekében. A begépelt jelszó nem látszik a képernyőn, és a hibátlan bevitel biztosítása érdekében kétszer kell begépelni. Ha újraindítjuk a rendszert, root néven és ezzel a jelszóval léphetünk be. Figyelem! Ha elfelejtjük a jelszót, nem tudunk belépni a rendszerbe, és ezen senki nem tud segíteni.

A root azonosítót csak konfigurálás, rendszerkarbantartás, másik user létrehozásának céljából érdemes használni. Ne indítsunk root-ként ismeretlen programokat, mert azok veszélyesek lehetnek. Ha betartjuk ezt, megkímélhetjük magunkat nagyon sok kellemetlenségtől, és a vírusok elterjedését is lehetetlenné tesszük a rendszerünkön. (Valójában emiatt nincsenek Linuxos vírusok.)

A LILO telepítése

A LILO (Linux Loader) egyrészt elengedhetetlen a Linux elindításához, másrészt egyben bootmanager, mellyel választhatunk a különféle operációs rendszerek, illetve a különböző Linux kernelek betöltése között. Több helyre is telepíthetjük, de vigyázzunk, hogy onnan valamilyen módon elindítható legyen. A legegyszerűbb megoldás, ha az IDE primary master (vagy - csak SCSI merevlemez esetén - a legelső) merevlemez MBR-ébe (Master Boot Record) kerül, és onnan a gép bekapcsolásakor automatikusan elindul. Ekkor felülírja az eddigi MBR-t, tehát az eddig használt rendszereket csak a LILO-n keresztül tudjuk elérni.

Tehetjük a LILO-t a Linux partíció elejére is. Ilyenkor kell egy külső bootmanager program, ha a második merevlemezre került a Linux. Ha az elsőre, akkor a Linux partíciót kell aktívvá tennünk.

További alternatíva, hogy floppyra tesszük a LILO-t. Ilyenkor a Linux csak e floppy segítségével indítható, de nem onnan indul. Azaz csak a LILO indul a floppyról. Ha nem sikerül a LILO helyes telepítése, a Linux installálásának elején létrehozott bootlemezt hívhatjuk segítségül. Azt használva is elindíthatjuk a telepített Linux rendszert, hogy kijavíthassuk a LILO beállítását. Fontos megjegyezni, hogy ilyenkor a Linux kernele a floppyról és nem a merevlemezről indul.

A Red Hat telepítőnek a LILO helyére vonatkozó első kérdésére a fentiek alapján kell válaszolnunk. A következő lépés a bootolandó rendszerek megadása. A frissen telepített Linux rendszer alapértelmezésben benne van, de felajánlja a telepítő az általa megtalált rendszerek bootolhatóvá tételét is a LILO-n keresztül.

Fontos, hogy a LILO csak az 1023. cilinder alatti területről tudja elindítani a Linux kernelt. Ezt vegyük figyelembe a telepítéskor. Ha csak IDE vezérlőnk van, akkor bármelyikre kapcsolt, akár master, akár slave merevlemezről indítható Linux. Ha van SCSI és IDE merevlemez is, akkor csak a primary IDE, illetve 0-s SCSI merevlemezről. Ha csak SCSI van, akkor akár a 0-ás, akár az 1-es SCSI merevlemezről indítható a Linux.

Az új Linux rendszer

Ezzel a Linux telepítése befejeződött. Vegyük ki a bootlemezt a meghajtóból, nyomjuk meg az [Enter]-t és várjuk meg, míg elindul a Linux rendszerünk. (A LILO beállításaitól függően kézzel kell erre utasítanunk, ha nem a Linux az alapbeállítás.)

Sikeres indulás után a *login:* - *localhost login:* - prompt jelenik meg, szöveges módban. Itt, mint azt már leírtuk, root néven és a már megadott jelszóval kell belépni. *Belépés után az első és legfontosabb dolog, hogy létrehozzuk saját, normál jogú azonosítónkat!*

3.1.6.2. Felügyeleti szolgáltatások

A Linux alapú rendszerekben (alapértelmezett biztonsági beállítások mellett) minden felhasználó láthatja és használhatja a feltelepített programokat, telepíteni vagy törölni csak a kiemelt felhasználónak (root) van lehetősége. Linux alatt a programok telepíthető változatait tartalmazó állományokat csomagoknak nevezzük. Alapvetően kétféle csomagkezelési szabvány létezik: az RPM (RedHat Package Manager, a Linux alapértelmezett csomagformátuma, talán a legelterjedtebb) és a DEB (Debian). Az *RPM* igazi előnye abban rejlik, hogy ellenőrizni tudja a függőségeket a programok között, ezért nem fogja megengedni, hogy olyan programot telepítsünk fel, amihez a szükséges könyvtárak nem állnak rendelkezésre.

Az Intel (és a vele kompatíbilis családba tartozó) processzorokon futó programokat telepítő csomagfájlokat a következő szabályrendszer alapján nevezik el:

név.verzió.i386.rpm

(Ha egy forrás – azaz nem lefordított, futtatható kódot tartalmazó – csomagról van szó, az src tag is megtalálható a névben.

Linux alatt két lehetőség van az RPM csomagok kezelésére: az „rpm" parancs (parancssorból), vagy a grafikus Kpackage program.

Az ***rpm*** parancssori változatában a csomag kezelésére vonatkozó művelet opcionális kapcsolók segítségével adható meg. A program általános alakja: „rpm -opciók csomag_neve". Az opciók közül fontosabbak:

- i: csomag telepítése. A programot a csomagkezelő feltelepíti, az azonnal használható. Általában kiegészítve használjuk a „v" (részletes leírás a telepítés menetéről) és a „h" (telepítés állapotának kijelzése) kapcsolókkal.
- u: program frissítése. A csomag telepítése során a program minden régi konfigurációs fájlról mentés készül.
- e: program eltávolítása.
- V: program ellenőrzése. Az ellenőrzés a feltelepített programcsomag valamennyi szükséges fájljának meglétét vizsgálja.
- q: információs kapcsoló. Hatására megjeleníthető a csomaghoz tartozó fájlok listája, a csomag rövid leírása, esetlegesen (további kapcsolókkal kiegészítve) a csomag telepítéséhez szükséges fájlok listája.

A **KPackage** egy grafikus csomagkezelő program, a rendszeren elérhető és telepíthető csomagok kezelésének eszköze. Felülete két ablakból áll: a bal oldalon a csomagok listája tárgykör szerint rendezve, a jobb oldalon a kiválasztott csomagról szóló információk láthatók. A fájlnál lévő Lista fül lehetővé teszi a csomagot alkotó fájlok megjelenítését.

Csomag telepítése történhet közvetlenül a KFM fájlkezelőből is. Csomagfájlra kattintva a KPackage ablaka megjelenik és a Telepítés gombbal a csomag telepíthető. A másik módszer a KPackage használata. A Beállítások menüben megadható (a „Nem telepített RPM csomagok telepítése" parancs kiadása után) a telepíteni kívánt csomagot tartalmazó hely (lehet Internetes cím is!), majd a Fájl menüben megjelenő listában a megfelelő csomagok kiválasztásával történhet a telepítés. Egy csomag törléséhez ugyanitt elegendő a csomagot kiválasztani, és az Eltávolítás gombra kattintani.

A **Feladatkezelő** (a Rendszer menü alatt található) lehetővé teszi az éppen futó folyamatok listájának megjelenítését, illetve a futó folyamatok állapotába történő beavatkozást. (Ugyanezeket az információkat nyújtja **Kpm** program is - a Segédeszközök

menü alatt található –, csak egy kicsit más formában.) A KDiskFree megjeleníti a csatlakoztatott háttértárak foglaltságát (százalékban kifejezve).

A **SysV Init** szerkesztő egy program, amely által kiválaszthatók a háttérben futó démonok. (Csak root joggal használható!) A bal oldali ablakban a telepített szolgáltatások jelennek meg, a többi ablakban pedig felül azok, amelyek elindulnak, alul azok, melyek leállnak az adott futási szinten. Mindegyik futási szint külön látható. Például a 6-os szinten (gép újraindítása) a reboot-on kívül minden szolgáltatás leáll. Említést érdemel még a 3-as (alap szint) és az 5-ös (grafikus szint) futási szint is.

3.1.6.3. Linuxconf

A *Linux* rendszer igen sok szolgáltatást és egyéb segédeszközt tartalmaz, amelyek viselkedése és aktivitása nagy számú konfigurációs fájlon keresztül állítható be. E fájlok egyszerű, szöveges formátuma könnyű olvashatóságot biztosít (legalábbis az angolul beszélő felhasználók számára). A konfigurációs fájlok legnagyobb része a /etc könyvtárban található.

A *linuxconf* program segítségével grafikus felületen keresztül állítható valamennyi konfigurációs fájl tartalma. Használatával precíz módon és igen könnyen (egérrel) szabályozható a *Linux* viselkedése. A *linuxconf* moduláris felépítésű: az alapértelmezés szerint részét képező modulok mellett további felügyeleti modulok hozhatók létre (vagy tölthetők le az Internetről), így tetszés szerinti (egyéni) rendszer-felügyeleti eszközzé alakítható. A *linuxconf* ezeken felül környezetfüggetlen: van szöveges felülete (ami a konzolon futtatható), grafikus felülete és HTML felülete is.

A *linuxconf* használatához „root” felhasználói jogosultság szükséges. Indítása történhet a „linuxconf” paranccsal, vagy a K menü Rendszer menüjén keresztül.

Hálózati beállítások

A hálózati beállítások minden *Unix* alapú rendszerben kulcsfontosságú szerepet játszanak, ez alól a *Linux* sem kivétel. A *linuxconf*-nak vitathatatlan erénye, hogy egy átlátható és jól szervezett felületet ad a legtöbb hálózati beállítás elvégzéséhez.

Az alapbeállítások közé a számítógép nevének és a használt hálózati interfésznek (általában Ethernet kártya) a megadása tartozik. A gépnév (alapértelmezés szerint) a „teljes tartománynév” (FQDN), amit a telepítés során kell megadni.

A hálózati kártya beállításai nem nélkülözhetnek bizonyos alapvető ismereteket az IP alapú hálózatok felépítéséről, működéséről, azonosítási rendszereiről és az általános hálózati protokollok szerepéről[18]. Ha a címkiosztás *DHCP* vagy *BOOTP* protokoll alapján történik, akkor nem sem az IP-címet, sem a gépnevet nem kell megadni. Egyébként ki kell tölteni mind az IP-cím (és az alhálózati maszk), mind a géphez tartozó teljes név mezőket (és esetlegesen megadható egy hivatkozási „álnév” („alias”) is. Az interfész típusa függ a csatlakozó eszköztől: *ethn*, ha Ethernet hálózati kártya van a gépben (az „n” az kártya csatlakozási száma), *arcn*, ha *ARCnet* a kártya típusa, stb. A DNS megadásakor lényeges, hogy a *tartománynév* helyett a *névkiszolgáló(ka)t* kell megadni IP-címmel (nem a neveket!).

Az útválasztás panel segítségével a más számítógépekhez való csatlakozáshoz szükséges hálózati kapcsolat kiépítésének eszközei („routing”) adható meg. Amennyiben a számítógép saját maga működik útvonal-választóként („router”), úgy a „routed” démon konfigurálása is szükséges.

[18] Ezekkel az ismeretekkel a „Kommunikációs hálózatok” c. tantárgy részletesen foglalkozik, itt csak egy egyszerű konfiguráció kialakításához szükséges alapfogalmakat mutatjuk be!

Felhasználói fiókok

A *linuxconf*-nak ez a modulja lehetővé teszi a felhasználói azonosítók („accounts”) kezelését. (A hagyományos értelemben vett felhasználói fiókok mellett kezeli az ún. „speciális azonosítókat” is (ilyenek az egyes rendszerszintű szolgáltatások azonosítói), valamint használható különböző álnevek („alias”), és a jelszó-ellenőrzési szabályok felügyeletére is.)

A felhasználók megkülönböztetése egy egyedi azonosító alapján történik („user ID”, röviden „uid”). (Ajánlatos az 500 alatti uid-eket békén hagyni: mindegyiknek van már előre definiált szerepe.) Az egyes felhasználókat azonban csak a rendszer azonosítja az uid-dal, hivatkozásuk szöveges azonosítójuk segítségével (felhasználói név) történik.

Új felhasználó létrehozásához („*New user*”) a felhasználói név („*user name*”) mellett a felhasználó teljes nevét („*full name*”) is meg kell adni (ez utóbbi az azonosítást megkönnyítő, leíró jellegű információ – a rendszer működése szempontjából nincs különös jelentősége). A felhasználó jelszavának („*password*”) megadására vagy megváltoztatására a szokásos kétszeres ellenőrzéses („*confirm password*”) módszer szolgál.

A felhasználó létrehozása során megadható a felhasználó alapértelmezett környezete is: a „*login shell*” az alapértelmezett shell kiválasztására, „*home directory*” a $HOME helyének a megadására szolgál. A „*tasks*” az adott felhasználó nevében automatikusan elinduló programok ütemezésének az eszköze.

Megadható még csoport-azonosító is, ennek hiányában a rendszer létrehoz egy új csoportot, aminek csak az új felhasználó a tagja.

Az „*account info*” részben a fiókra vonatkozó általános beállítások tehetők, mint pl. a felhasználói fiók zárolása („*lock*”, az ilyen jelzéssel rendelkező felhasználó nem jelentkezhet be a rendszerbe a zárolás feloldásáig), vagy az érvényesség („*expiration date*”).

A felhasználói csoportok kezelésének alapelvei megegyeznek a felhasználóknál megismertekkel. A csoportok (ahogy azt az állományokkal kapcsolatos jogok bemutatásakor láthattuk) elsősorban a fájlok elérésének szabályozására szolgálnak. Egy felhasználó tartozhat egy vagy több csoportba is.

Fájlrendszerek

A *linuxconf* ezen moduljának legfontosabb tulajdonsága, hogy nagy mértékben leegyszerűsíti a háttértárak kezelését (legalábbis a „mount” parancs használatához képest). A modul elsődlegesen a háttértárak adminisztratív műveleteinek elvégzését támogatja akár helyi, akár távoli számítógépben elhelyezkedő tárolóeszközről legyen szó. Az alapvető műveletek (pl. a háttértár egy partíciójának fel- vagy lecsatlakoztatása) elvégzéséhez természetesen itt is tisztában kell lenni a Linux fájlrendszerének alapfogalmaival, mint pl. az eszköz (fizikai) azonosítója, a fájlrendszer típusa vagy a csatolási pont fogalma. A fájlrendszer típusa alapvetően meghatározza mind a tárolási szerkezetet, mind az adott háttértáron elhelyezkedő állományok és könyvtárak tulajdonságait (pl. FAT alapú fájlrendszerekben nem alkalmazhatók a speciális állománytulajdonságok vagy a lemezkvóták) – ennek következtében pedig az adott háttértárral a linuxconf-ban elvégezhető beállításokat is.

A fentiek (természetesen) kiragadott példák: a *linuxconf* ennél sokkal többre képes és alkalmas. Használata a modulok megismerésén (és az egyes modulok leírásának tanulmányozásán: ezeket nevezi a Linux-terminológia „How-To” („hogyan csináljam”)-nak) keresztül sajátítható el, és habár elsődlegesen felügyeleti (rendszergazdai) eszköz, ismerete az egyfelhasználós rendszerben is hasznos lehet.

Összefoglalás

A fentiek természetesen nem nyújthatnak teljes és részletes képet a Linux alapú rendszerek valamennyi szolgáltatásáról, de reményeink szerint elegendő alapot adnak az önálló munka megkezdéséhez. Ne feledje: az elsajátított ismeretek akkor hasznosulnak, ha azokból gyakorlatban is alkalmazható készséget fejleszt! A Linux alapú operációs rendszerekkel szemben a rendszer általános leírásában említett ellenérzet az utóbbi időben egyre inkább csökkenni látszik, az ilyen operációs rendszerek elterjedtsége pedig növekvő tendenciát mutat. Megismerésével egy korszerű informatikai eszköz használatának képességével gazdagodott. Gratulálunk!

Ellenőrző kérdések

1. Soroljon fel 3-4, a UNIX/Linux alapú operációs rendszerekre jellemző tulajdonságot!
2. Soroljon fel UNIX/Linux alapú operációs rendszereket!
3. Mit jelent a Linux esetében a disztribúció? Miben áll a gyakorlati jelentősége?
4. Mit jelent a UNIX/Linux alapú rendszerek esetében a rétegszemlélet?
5. Magyarázza meg a következő fogalmakat: process, kernel, shell, uid!
6. Ismertesse a gyökér, (/), a /usr és a /var fájlrendszerek szerkezetét, szerepét!
7. Milyen állománytulajdonságokat kezel a Linux?
8. Hogyan működik a Linux alapú rendszerekben a fájl-szintű jogosultsági rendszer?
9. Ismertesse a nyomtatás hagyományos menetét támogató rendszereszközöket Linux alatt!
10. Soroljon fel 3-4 Linux parancsot! Csoportosítsa őket a működésük jellege szerint, az egyiket mutassa be részletesen!
11. Jellemezze a megismert grafikus felületet!
12. Soroljon fel olyan alkalmazásokat, amelyekkel Linux alapú rendszerekben találkozhat!
13. Melyek a rendszerfelügyelet alapvető eszközei Linux alatt?
14. Mi a csomag? Hogyan történhet a csomagok kezelése Linux alatt?
15. Jellemezze a linuxconf felügyeleti eszközt! Mit jelent a modularitás és hogyan valósul meg az esetében?
16. Milyen előkészületeket kell elvégezni egy Linux rendszer telepítését megelőzően?
17. Milyen lépésekben valósul meg egy Linux rendszer telepítése?
18. Miért célszerű (kell) a telepítést követően újabb felhasználót létrehozni (a root mellett)?

3.2. Microsoft Windows operációs rendszerek

A Microsoft és Bill Gates az elmúlt 25 évben ugyanolyan meghatározó tényezővé vált a számítástechnikában, mint korábban Hollerith vagy Neumann. A Windows (minden hibájával és számos kritikájával együtt is) töretlen sikere és elterjedtsége megköveteli, hogy foglalkozzunk ezzel az operációs rendszerrel is. Azonban mielőtt elkezdenénk a részletes ismertetést, szeretnénk leszögezni, hogy a leírtak a jegyzet készítésének idején elérhető verziók sajátosságait tükrözik, különös tekintettel a gyakorlati módszerekre, amelyek a jelenleg utolsó hivatalosan kiadott változatnak, a Microsoft Windows XP-nek a specifikumait tárgyalják. Szeretnénk továbbá emlékeztetni a hallgatót, hogy a tankönyv ezen fejezete ugyan elsődlegesen a gyakorlati kérdésekre koncentrál, de sikeres feldolgozásához elengedhetetlen az első részben leírt elméleti alapfogalmak és módszerek ismerete.

3.2.1. W, mint Windows

A Microsoft eredetileg a DOS operációs rendszerrel alapozta meg mind hírnevét, mind piaci pozícióját – sőt, a Windows operációs rendszer alapgondolatát is. A DOS ugyanis egy egy-felhasználós, kötegelt módú, karakteres felületű operációs rendszer volt – viszont rendkívül sikeres. A ’80-’90-es évek közepétől számos olyan alkalmazás megjelent (menüvezérelt (de még karakteres felületű) rendszerek a parancskiadás egyszerűsítésére: commanderek; grafikus felületű rendszerek: X11, XEROX, Lisa), amelyek a DOS nehézkességén próbáltak segíteni. Ebbe a fejlesztési folyamatba kapcsolódott be a Microsoft is, amikor 1985-ben megjelentette saját grafikus felületű kezelőprogramját, amelyet Windows-nak nevezett el. A Windows sikeres volt ugyan, de igazi áttörést csak a harmadik változata (Windows 3.0) hozott (1990-ben, majd a 3.1 (1992), illetve a 3.11 (1994)). A sikeren felbuzdulva (és jó üzleti érzékkel nyitva a hálózati rendszerek irányába) 1993-ban megjelent a Windows NT („New Technology”) – a Windows vállalati környezetre szánt, robosztusabb, körülményesebb, de megbízhatóbb rendszere. Ez a döntés viszonylag hosszú időre megosztotta a Microsoft fejlesztői kapacitását, így a következő rendszerek ennek a kettősségnek (nevezetesen, hogy külön operációs rendszer(ek) jelentek meg az otthoni felhasználók, és külön rendszerek a vállalati igények kiszolgálására) jegyében jelentek meg.

1995-től a Microsoft-nál a verziószámot (egy időre) felváltja az operációs rendszerek az azonosítására a megjelenés (tervezett) évszáma: az otthoni felhasználók a Windows 95 és a Windows 98, a vállalati környezetben dolgozók a Windows 2000 megjelenésekor szembesültek ezzel - közben (1996-ban) megjelenik „Compact Edition” néven a hordozható (kézi)számítógépekre („palmtop”) optimalizált Windows verzió is. (A Windows 2000 megjelenésével az eddigi kettős fejlesztési rendszer tovább finomodik, legalábbis ami a hálózati kiszolgálókat (szervereket) illeti: a Windows 2000 Server operációs rendszer „család” különböző konfigurációkon (és igények esetén) más verziókban érhető el.) Szintén 2000-ben jelent meg az otthoni használatra szánt Windows verziók újabb képviselője „Millenium Edition” néven – talán ez a legtöbbet kritizált és legkevésbé elterjedt az eddig említett Windows verziók közül. 2002-ben a Windows XP („eXPerience”) megjelenésével a Microsoft újra összekapcsolja a két fejlesztési irányt: az új operációs rendszer egyszerre továbbfejlesztése a Windows 9x (asztali) és az NT/2000 (hálózati) operációs rendszereknek (olyannyira, hogy az XP megjelenésével a Windows 2000 felhasználói változata („Windows 2000 Pro”) el is tűnik a Microsoft palettájáról, másrészt a Microsoft az XP esetében is alkalmazza a verzión belüli megkülönböztetést: Home és Professional változatokban jelenik meg). 2003-ban a Windows 2000 szerverek továbbfejlesztéseként jelent meg a Windows 2003, és a Microsoft 2005-re tervezte (bár a jegyzet írásakor még nem jelent meg) az XP „utódát” (fejlesztési kódnevén „LongHorn”, az elérhető (alfa és béta kódú teszt-)verziókban „Vista”).

Az egyes verziók jellemzői

1. Windows 1.x – 2.x
    a. nem tekinthető önálló operációs rendszernek (grafikus felület a DOS-hoz)
    b. Win 2.1: processzor-architektúrához kötődő verziók: /286, /386
2. Windows 3.x
    a. kooperatív multitasking: az időosztásos elv szerint párhuzamosan futó folyamatok az operációs rendszer felügyelete nélkül (saját hatáskörükben) döntenek az időszelet lejártakor a CPU átadásáról – aminek következtében, ha az aktuális (a CPU-t birtokló) alkalmazás lefagy, akkor nem tudja a CPU-t továbbadni, így az összes többi (egyébként még futásra képes) alkalmazás is várakozó állapba kerül, amit a felhasználó úgy érzékel, hogy valamennyi lefagyott…
    b. 16 bites alkalmazások
    c. Win 3.11 (Windows for Workgroup): egyenrangú hálózatok (munkacsoport) támogatása
3. Windows NT
    a. önálló operációs rendszer (saját kernel)
    b. preemptív multitasking: a párhuzamosan futó folyamatok között a CPU kiosztása az operációs rendszer feladata
    c. 32 bites alkalmazások támogatása
    d. NTFS fájlrendszer
    e. kliens-szerver hálózati architektúra (tartomány) támogatása – ennek megfelelően kétféle verzió: NT Server és NT Workstation
4. Windows 95/98/ME
    a. (a Win95 esetében megoszlanak a vélemények, hogy valódi operációs rendszerről beszélhetünk-e, vagy még mindig egy DOS kernel grafikus felületéről van szó)
    b. FAT32: nagyobb tárolókapacitás, hosszú fájlnevek használata
    c. újratervezett felhasználói felület: Asztal, Tálca
    d. fejlesztett multimédia-támogatás, DirectX
    e. integrált Internet (amiért később az ún. „Antitröszt-per”-ben el is marasztalták a Microsoftot)
    f. verziók:
        i. Win95: OSR-1, OSR-2 (javított kiadás), Plus for Win95 (multimédia jellegű szórakoztató kiegészítések és bővítmények)
        ii. Win98: Win98, Win98 SE („Second Edition”, az első verzió hibáinak a javítását tartalmazó kiadás, nem javítócsomag, hanem önálló verzió!)

5. Windows 2000/XP
    a. központosított hálózati nyilvántartási rendszer (LDAP alapú címtár: Active Directory)
    b. NTFS v5: titkosító fájlrendszer (EFS), lemezkvóták, dinamikus kötetek
    c. verziók
        i. Win2000 (egyes források szerint az egyes változatok ugyanazon a kernelen alapulnak, de funkcionális korlátozásokkal rendelkeznek):
            1. Professional: munkaállomások, asztali számítógépek igényeire optimalizált, max. 2 processzor – az XP megjelenésével szerepét gyakorlatilag az XP Professional vette át
            2. Server: vállalati kiszolgálói szerepre optimalizált, max. 4 processzor
            3. Advanced Server: nagy megbízhatóságú kiszolgáló, fürtözhető, max. 8 processzor
            4. Datacenter: nagy méretű rendszerek tranzakciós kiszolgálója, max. 32 processzor
        ii. WinXP: Starter, Home és Professional verziókban jelent meg (a Starter csak a Távol-Keleten került forgalomba), javítócsomagjai közül legfontosabb az SP2 (amely a megjelenés óta felismert hibákon kívül számos biztonsági kiegészítést is tartalmaz)
6. Windows 2003
    a. .NET integráció
    b. csak szerver verziója van, a Win2000-hez hasonlóan négy (igény szerint optimalizált) architektúrában érhető el: Standard Edition (1-10 felhasználó), Web Edition (webszerver szolgáltatások), Enterprise Edition (vállalati kiszolgáló, 25-nél több felhasználó), Datacenter Edition

A jegyzet írásának időpontjában már sokat lehet hallania és olvasni a következő Windows verzióról (a legvalószínűbb, hogy „Windows Vista” név alatt fog megjelenni), de egyelőre csak a teszt-verziók érhetők el. A leírások tanúsága alapján a Vistában ismét számos újdonság jelenik meg, a legérdekesebbnek az NTFS kiterjesztéseként beharangozott új fájlrendszer tűnik (lehetővé válik többszörös hivatkozások és összetett keresések alkalmazása), de a felületen megjelenő újdonságok (3D Asztal: Aero, interaktív másodlagos tálca: „sidebar”, stb.) is bizonyára sokakat káprázтatnak majd el. De mielőtt elmerülnénk a legutolsó, elérhető változat felületének és GUI megvalósításának részleteiben, tekintsük át röviden a Windows karakteres felületének parancsait!

3.2.2. A karakteres felület használata

Ahogy az a fenti felsorolásból is kitűnik, a Windows erősen kötődik és épít a grafikus felületre, de ez nem jelenti azt, hogy ne rendelkezne karakteres felületen is elérhető utasítás-rendszerrel. Jogosan merül fel a kérdés, mely szerint a grafikus felület eszközrendszere és az operációs rendszer szolgáltatásainak bősége mellett mi szükség lehet utasításokkal beavatkozni a rendszer működésébe? A válasz lehet(ne) alapvetően szemléletet tükröző is: a karakteres felületen való műveletvégzés gyorsabb, mint a grafikus környezet használata – de véleményem szerint az utasítások ismerete elsődlegesen felügyeleti tevékenységek

megvalósításakor hasznos: pl. a Rendszer-helyreállítási konzol – mint a megsérült, de valamilyen oknál fogva nem újratelepíthető rendszer megmentésének utolsó lehetséges eszköze – használata nem képzelhető el az utasítások ismerete nélkül. A következőkben a teljesség igénye nélkül a legfontosabb utasításokkal ismerkedünk meg.

A karakteres felület többféle módon is elérhető (az operációs rendszer számára valójában nem felületről, csak egy újabb alkalmazásról van szó): alapértelmezett telepítés mellett megtalálható az elindítására szolgáló parancsikon („Parancssor") a START menü Programok menüpontjának Kellékek csoportjában, de elindítható a Futtatás menüponthoz begépelt cmd vagy command paranccsal is – ez utóbbi kettő között az alapvető különbség, hogy míg a cmd a WinXP parancsértelmezője, a command a DOS-szal való kompatibilitás biztosításának az eszköze.

Az XP utasítás-szerkezete megfelel az operációs rendszerek általános alapelveinél leírtaknak: parancsnév paraméter(ek) /kapcsoló(k). A következőkben a leggyakrabban előforduló parancsok rövid áttekintését adjuk:

1. lemezkezelés parancsai
    a. FORMAT – megformázza a kijelölt háttértárat: új gyökérkönyvtárat (és fájlrendszert) hoz létre (az esetleges előző fájlrendszer és az abban létrehozott állományok természetesen törlésre kerülnek), illetve ellenőrzi a megadott háttértár szektorait. Nem használható rendszert tartalmazó köteten. A helyreállítási konzolból indítva megadható a létrehozandó lemez fájlrendszere is.
    b. CONVERT – FAT vagy FAT32 fájlrendszerű kötetet NTFS kötetté alakít.
    c. CHKDSK – lemezellenőrzést hajt végre a megadott köteten: megvizsgálja a szektorokat, megjelöli a hibás szektorokat (és megpróbálja helyreállítani a bennük található adatokat – sérült lemez esetén célravezetőbb lehet a RECOVER parancs használata).
    d. LABEL – létrehozza, megváltoztatja vagy törli egy meghajtó logikai azonosítóját (kötetcímkéjét). A kötetcímke aktuális értékének lekérdezésére használható a VOL parancs is.
2. könyvtárakra (mappa) vonatkozó parancsok
    a. MKDIR (MD) – létrehozza a megadott könyvtárat (könyvtár-szerkezet megadása esetén a tartalmazó könyvtárak is létrejönnek!).
    b. CHDIR (CD) – megjeleníti vagy kiválasztja (módosítja) az aktuális könyvtárat (könyvtár-váltás).
    c. RMDIR (RD) – törli a megadott könyvtárat. Az aktuális könyvtár vagy annak „szülő"-könyvtára, továbbá nem üres könyvtár nem törölhető!
    d. TREE – grafikus formában megjeleníti a megadott könyvtárból kiinduló könyvtárszerkezetet.
    e. SUBST – virtuális (logikai) meghajtót hoz létre a megadott elérési út, mint gyökérkönyvtárból.
3. állományokra vonatkozó parancsok
    a. COPY – állományokat másol. A parancs általános alakja: COPY forrás cél, ahol általában a „forrás" és a „cél" is elérési út, de „forrás"-ként a billentyűzetet megadva – logikai eszközazonosítója „CON" – szöveges állomány létrehozására is alkalmas, pl: COPY CON level.txt. Alkalmas állományok összefűzésére is.
    b. MOVE – állományokat helyez át.
    c. RENAME (REN) – állományok azonosítóját (név, típus) változtatja meg.

d. ATTRIB – beállítja, módosítja, megjeleníti vagy törli az állományok karakteres felületen elérhető jellemzőit (attribútumait). (A CACLS parancs hasonló műveleteket végez el az állományok jogosultsági jellemzőivel kapcsolatban.)
e. TYPE – egy (szöveges) állomány tartalmát megjeleníti.
f. DEL (ERASE) – állományokat töröl. Figyelem: a karakteres felület törlési művelete nem logikai – a törölt fájlok ténylegesen és véglegesen eltávolításra kerülnek (és nem a Lomtárba…)!

4. könyvtárakra és állományokra egyaránt alkalmazható parancsok
   a. DIR – állománytulajdonságok megjelenítése: név, típus, időbélyegek, stb.
   b. COMPACT – (csak NTFS partíción) a röptömörítés szolgáltatás használatának beállítása, lekérdezése.
   c. XCOPY – teljes könyvtárszerkezet (állományok és könyvtárak is!) másolása.
5. egyéb parancsok: környezet lekérdezésének, beállításának parancsai:
   a. DATE, TIME – rendszerdátum és idő
   b. VER – az operációs rendszer verziószáma
   c. PROMPT – a készenléti jel alakja (alapértelmezés szerint az aktuális könyvtár teljes elérési útja és egy „>” jel)
   d. PATH – keresési útvonalban szereplő könyvtárak (az ezekben a könyvtárakban található végrehajtható kódot tartalmazó állományok a könyvtárszerkezet bármely pontjáról elindíthatóak)
   e. HELP – a karakteres felület parancsainak rövid leírása (bármely parancs neve után a „/?” kapcsoló az adott parancsra vonatkozó részletes leírást jelenít meg)
   f. EXIT – a cmd parancsértelmező befejezése, visszatérés a grafikus felülethez.
6. a helyreállítási konzolban használható fontosabb parancsok
   a. a parancsok között vannak olyanok, amelyek elérhetőek a karakteres felületen – normál működés mellett – is, azonban az egyes parancsoknak vannak olyan kapcsolóik, módosítóik is, amelyek csak a helyreállítási konzolban használhatók: ATTRIB, CHDIR, CHKDSK, COPY, DELETE, DIR, EXIT, FORMAT, MKDIR, RENAME, RMDIR, TYPE.
   b. DISABLE, ENABLE – letilt (leállít) vagy engedélyez (elindít) egy rendszer-szolgáltatást vagy egy eszközillesztőt.
   c. DISKPART – a partíciók kezelésének eszköze.
   d. EXPAND – kibont egy fájlt egy tömörített fájlból (a Windows alapértelmezett tömörítési formátuma a kabinet (.cab), a telepítőkészleten a Windows komponensei (úgy az alkalmazások, mint pl. az eszközvezérlő (illesztő, „driver”) programok) jelentős része ilyen fájlok formájában érhető el.
   e. BOOTCFG, FIXBOOT, FIXMBR – beállítja vagy helyreállítja a rendszert indító fájlt (boot.ini), a partíciós tábla tartalmát vagy a lemez fő rendszertöltő rekordját („Master Boot Record”).
   f. SYSTEMROOT – az operációs rendszert tartalmazó könyvtár (alapértelmezés szerint C:\Windows) helyének megváltoztatása.

Ahogy az a fenti (közel sem teljes) felsorolásból is látható, a Windows karakteres felülete ugyanolyan hatékony (sőt, bizonyos esetekben – pl. hálózati paraméterek konfigurálása esetén – még jobban használható) eszköz a felhasználó (vagy a rendszergazda)

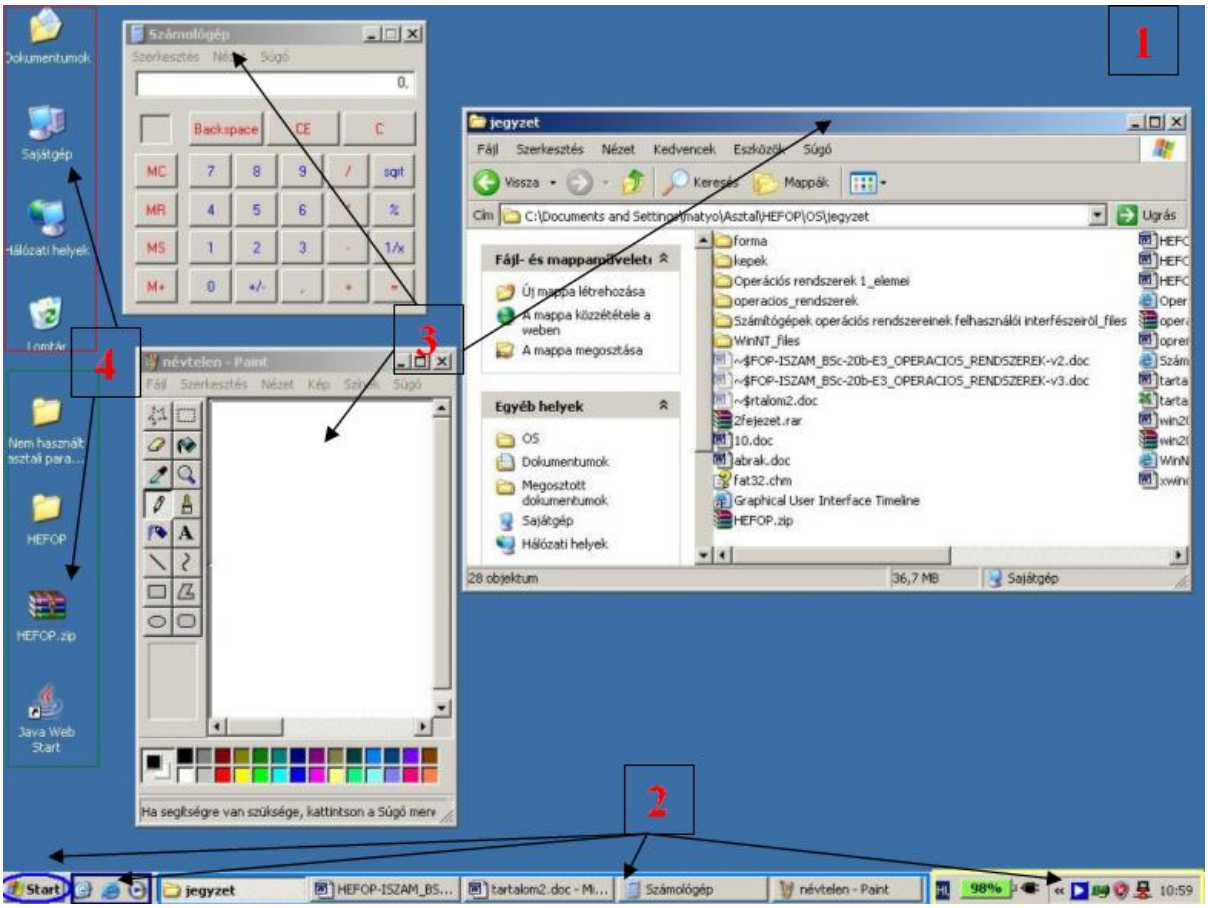
munkájában, mint a grafikus környezetben. Ezzel együtt a Windows eredendően egy grafikus felületű operációs rendszer, a következőkben tekintsük át ezeket a komponenseket is!

3.2.3. A grafikus felület felhasználói eszközei

Ahogy azt már az X Window System tárgyalásakor említettük, a Windows (legalábbis az X11 fogalmai szerint) a szabvány által ablakkezelőnek nevezett felügyeleti eleme a GUI-nak. A Microsoft fogalmai szerint a felhasználói munkaterület az Asztal – felfogható legmagasabb („root”) szintű ablaknak is. Az Asztal megjelenési formája (ahogy az Asztal a felhasználó számára megjelenik) a Tapéta (vagy Háttér). Az Asztal valamelyik szegélyénél helyezkedik el a Tálca, ami lényegében parancsikonok gyűjteménye. Az Asztalon alkalmazások (a hozzájuk tartozó Ablakban) illetve ikonok helyezhetők el.

3.2.3.1. Az Asztal szabványos elemei

A 3-1. ábra az Asztal szokásos elrendezését és szabványos komponenseit szemlélteti.



3-19. ábra: A Windows grafikus felülete

Az Asztal a felhasználói munkavégzés eszköze (3-12. ábra, 1). A Tapéta az Asztalon megjelenő grafikus elem, lehet egyszínű kitöltés (az 3-12. ábrán a Windows XP alapértelmezett kék színe látható), vagy valamilyen grafikus formátumú állomány tartalma (ez utóbbi esetben a kép méretének és a grafikus felbontásnak (az Asztal „méretének”) függvényében különböző megjelenítési módok alkalmazhatók: átméretezés (a két méret kiegyenlítése a kép méretének arányos nyújtásával vagy zsugorításával), megjelenítés (a kép saját méretében az Asztal közepén helyezkedik el), ismétlés (a kép az Asztal bal felső sarkától kiindulva, amennyiben kisebb az Asztalnál, akkor jobbra és lefelé ismételve jelenik meg). Az Asztal kinézetével kapcsolatban a Windows XP számos előre definiált összeállítást tartalmaz, amibe nem csak a Tapéta „képe” tartozik bele, hanem az Asztalon megjelenő valamennyi

(ablak)komponens formai leírása is – ezek a Sémák –, de ezek a megjelenítési beállítások (akár elemenként is) egyénileg is kialakíthatók.

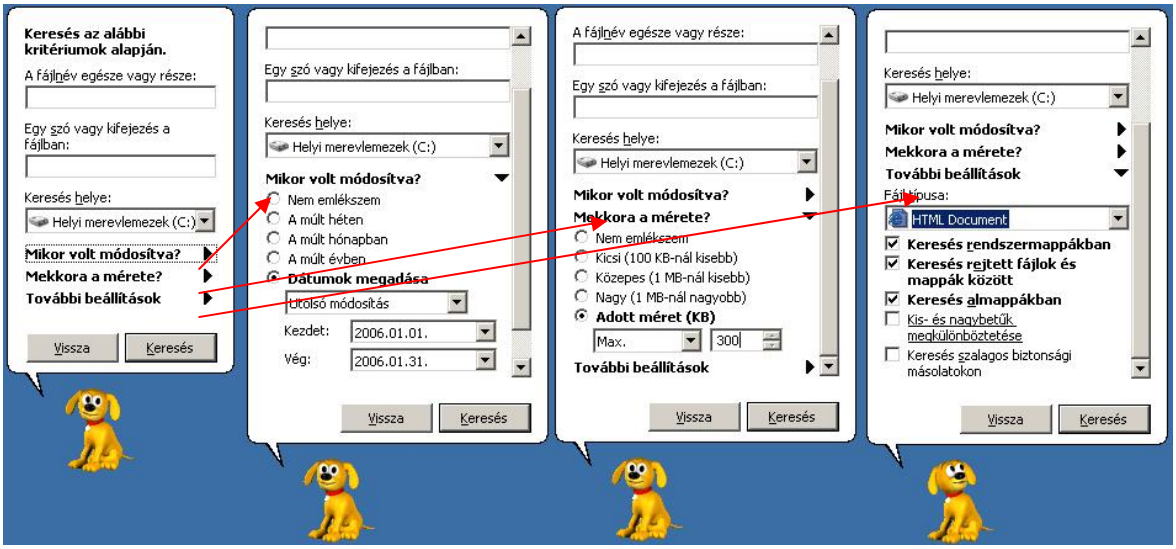
A Tálca a Windows kezelésének alapvető vezérlési eszköze (3-12. ábra, 2). Alapértelmezés szerint az Asztal alsó szegélyénél helyezkedik el, de bármelyik szegélyre (alsó, felső, bal, jobb) áthelyezhető, továbbá mérete is változtatható (legfeljebb az Asztal területének feléig növelhető, illetve akár teljesen el is rejthető). A Tálca részei (balról jobbra): START menü, gyorsindító eszköztár (lényegében a START menü Programok elemének kivonatolt változata: a felhasználó által leggyakrabban használt programok elindítására szolgáló ikonok), alkalmazás-ablakok ikonizált állapotának megjelenítési területe (a futó programokhoz tartozó ablakok ikon méretű megfelelői, lehetővé teszi az alkalmazások közti váltást, a Windows XP újdonsága, hogy az azonos alkalmazáshoz tartozó ablakokat képes egyetlen ikon-csoportban megjeleníteni), státusz (állapot-jelző) terület (a rendszerszolgáltatások és a háttérben futó felhasználói folyamatok (azaz olyan programok, amelyekhez nem tartozik az Asztalon ablakterület) ikonjainak gyűjteménye, amelyen keresztül ezek az alkalmazások felügyelhetőek).

A START menü a Windows alapértelmezett kezelési eszköze. Parancsikonok csoportosított (és szervezett) gyűjteménye, amelyben egyaránt megtalálhatók az operációs rendszer felhasználói szolgáltatásaira vonatkozó elemek (felhasználói munka befejezésével kapcsolatos feladatok, beállítási műveletek, támogatási szolgáltatások, rendszerkomponens alkalmazások) és a felhasználó munkavégzéséhez kapcsolódóan létrejövő saját elemek (utoljára használt állományok – illetve alkalmas felületi beállítás esetén programok is – jegyzéke, felhasználó által telepített alkalmazások (programok) elindítására szolgáló parancsikonok). Az egyes elemekkel kapcsolatban a következőket érdemes tudni:

1. leállítás: a felhasználói tevékenység befejezésének szabályos eszköze (a számítógép - fizikai szintű – kikapcsolása az operációs rendszer megfelelő leállítása nélkül adatvesztést, illetve a rendszerállományok sérülését okozhatja!). A számítógép leállításával kapcsolatban több lehetőség is elképzelhető:
   a. a leállítás a számítógép fizikai kikapcsolását eredményezi (régebbi típusú alaplapok esetén csak lehetővé teszi);
   b. az újraindítás az operációs rendszer teljes újratöltését eredményezi (mintha csak most kapcsoltuk volna be a számítógépet);
   c. a kijelentkezés az adott felhasználó és a hozzá kapcsolódó rendszerszolgáltatások befejezését valósítja meg (alapvető különbség az újraindításhoz képest, hogy nem a teljes rendszerbetöltési folyamat ismétlődik meg, csak a bejelentkezéstől a felhasználói környezet kialakításáig terjedő rész, azaz nem minden operációs rendszer szolgáltatás indul újra);
   d. a felhasználó-váltás anélkül teszi lehetővé más felhasználói környezetre (és munkamenetre) történő áttérést, hogy az adott felhasználó tevékenysége(i) befejeződne(nek) – a jelenlegi felhasználói környezet működése mellett egy új felhasználói környezet indítására ad lehetőséget;
   e. a készenléti állapot és a hibernálás pedig a munkamenet felfüggesztésének (energiatakarékos) eszközei, mindkét esetben a kijelölt állapotból visszatérve a felhasználói tevékenység a felfüggesztést megelőző állapotból folytatható (azaz a felület, a futó programok, stb. ugyanaz lesz, mint a felfüggesztés előtt). Alapvető különbség a két leállítási lehetőség között, hogy készenléti állapotban csak alacsonyabb energiaszintre kapcsolódnak a számítógép hardver egységei (a nem használt eszközöket az operációs rendszer lekapcsolja), hibernálás

esetén pedig a teljes környezet (a felület, a programállapotok, a memóriatartalom, stb.) lementésre kerül a háttértárra. (Ebből következően készenléti állapotban egy esetleges áramszünet adatvesztést okoz, hibernálás esetén nem, másrészt a hibernálás használatához szükséges a fizikai memóriaméretnek megfelelő szabad tárhely valamelyik háttértáron.)

2. futtatás: a felhasználói parancskiadás eszköze (lényegében felfogható egyetlen parancs kiadására alkalmas karakteres felületnek is). Használható olyan alkalmazás elindítására, amely nem rendelkezik parancsikonnal (ilyenek pl. a karakteres felület parancsai, de ilyen a (felügyeleti eszközök között tárgyalásra kerülő) rendszerleíró adatbázis kezelő programja (regedit), vagy a Felügyeleti Konzol (mmc) is), vagy egy parancsikonnal rendelkező alkalmazásnak (az ikonban rögzítettektől) különböző paraméterekkel történő indítására.
3. súgó: az operációs rendszer felépítésével, működésével, kezelésével, hibaelhárításával kapcsolatos tudnivalók több szinten kategorizált, tematikusan és kulcsszó alapján egyaránt kereshető rendszere.
4. keresés: a számítógépen (illetve hálózati környezetben az adott hálózati szegmensben) elérhető információk többszintű keresését lehetővé tevő eszköz. Legáltalánosabban állományok keresésére használható, a háttértáron elhelyezkedő állományok helyének meghatározásának eszközrendszerét bővíti: (a logikai tárolási rendszerben – könyvtár-struktúrában – elfoglalt hely megadása mellett) további keresési ismérvek kezelésére is képes. A legfontosabbak a tartalom szerinti keresés, illetve az állományazonosító részleges ismeretében használható (helyettesítéses) keresés – ez utóbbi esetben az állomány ismert részeit helyettesítő- (maszk, joker) karakterekkel: „*” (tetszőleges számú, tetszőleges karakter) vagy „?” (pontosan egy darab tetszőleges karakter) egészíthetik ki –, de szinte az összes állománytulajdonságra (típus, méret, időbélyegek) alkalmazható (3-13. ábra).



3-20. ábra: A kereső különböző szolgáltatásai

5. beállítások: a rendszer működési paramétereinek módosítását lehetővé tevő (egyik, ld. a „felügyeleti eszközök” c. részt) felhasználói eszközök gyűjteménye (Vezérlőpult).
6. dokumentumok: ilyen néven két objektum szerepel a Windows alapértelmezett felhasználói felületén, némiképp hasonló szerepben, de alapvetően eltérő

szerkezettel: a START menü Dokumentumok eleme a felhasználó által utoljára szerkesztett állományokra vonatkozó hivatkozások jegyzéke, a Dokumentumok mappa (az Asztalon) pedig tényleges állományokat tartalmaz.

7. programok: az operációs rendszer és a felhasználó által telepített alkalmazásokhoz tartozó parancsikonok gyűjteménye.

Az ablakok (3-12. ábra, 3) a felhasználói tevékenységeket megvalósító alkalmazásokhoz rendelt képernyő-részek. Az ablakok általános szerkezete (címsor: az ablak által reprezentált program és (esetlegesen) a program által feldolgozott adatállomány ikonja és/vagy neve; menü; eszköztárak (az alkalmazáshoz kapcsolódó tevékenységekhez rendelhető ikonok); munkaterület, gördítősávok; állapotsor; keret) minden alkalmazás esetében azonos. Az ablakok kezelésével kapcsolatban az alapvető műveletek (mozgatás, alapértelmezett méretezési műveletek:  (kicsinyítés ikonméretre),  (az ablak az Asztal teljes felületét lefedi),  (váltás a teljes és az előző méret között)) a címsoron keresztül valósíthatók meg (az előzőektől különböző méret a szegély segítségével állítható be). Az inaktív ablakok címsora alapértelmezés szerint szürke, az aktív ablaké élénk (kék) – mindkét beállítás módosítható az Asztal megjelenítési sémáinak segítségével.

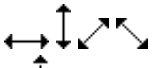
Az Asztalon alapvetően 3 (hálózati kapcsolat esetén 4, illetve 5) rendszerszintű ikon jeleníthető meg (5. ábra, 4): a Sajátgép (a Windows alapértelmezett állománykezelője), a Dokumentumok (rendszerkönyvtár, a felhasználó saját adatállományainak alapértelmezett tárolási helye), Lomtár (a (logikailag) törölt állományokra mutató hivatkozásokat tartalmazó speciális rendszerkönyvtár), továbbá (amennyiben a számítógép rendelkezik hálózati csatolóeszközzel (hálózati kártya, modem, stb.) a hálózatok kezeléséhez kapcsolódóan a „Helyi hálózatok” (az adott helyi hálózatban elérhető megosztott erőforrások áttekintésére, kezelésére szolgáló vezérlő) és alapértelmezett böngészőprogramként az „Internet Explorer”). A fentiek mellett a felhasználó szabadon helyezhet el az Asztalon újabb ikonokat (5. ábra, 4), amelyek alapvetően két típusúak lehetnek: közvetlen (egy, a háttértáron fizikailag létező objektumot reprezentáló) ikon, illetve közvetett (a Windows „parancsikonnak” nevezi ezeket az ikonokat még akkor is, ha az ikon által reprezentált – a parancsikon megjelenési helyétől független elem – nem futtatható parancs- vagy programállomány (alkalmazás), hanem adatfájl) – ez utóbbit az ikon bal alsó sarkában megjelenő nyíl alakú szimbólum jelzi.

A grafikus felület megjelenésével kapcsolatban fontos megjegyezni, hogy az operációs rendszer megjelenése a Windows filozófiája szerint szorosan kötődik a felhasználóhoz. Ez a gyakorlatban azt jelenti, hogy minden felhasználó saját felhasználói felülettel rendelkezik – ez alapértelmezés szerint természetesen azonos, hiszen ugyanazon rendszerszintű felhasználói felület másolataként jön létre, de a felhasználó saját beállításai ezt módosíthatják. A felhasználóra jellemző felület beállításai, további bizonyos szolgáltatások, alkalmazások működéséhez kötődő beállítások együttesen alkotják a felhasználói profilt. A profil alapértelmezett tárolási helye a rendszerkötet „Documents and settings” nevű (rendszer)könyvtárának a felhasználó nevével megegyező nevű mappája.

A Windows tartomány-alapú hálózati modellje lehetőséget biztosít (sok egyéb mellett) a profilok központosított kezelésére is: amennyiben a profilokat a tartomány (egyik) kiszolgálóján helyezzük el, akkor beszélhetünk központi vagy kötelező profilokról. A két profil működése hasonló abban az értelemben, hogy a bejelentkezés során a kiszolgálóról töltődik le és a munkaállomáson fejti ki hatását, és különbözik abban a tekintetben, hogy kötelező profil esetén a kijelentkezés során nem frissül a tartalma (azaz az ilyen profilok mindig ugyanazokat a beállításokat tartalmazzák, függetlenül a felhasználó tevékenységétől).

3.2.3.2. A felhasználói interakció eszközei

Grafikus felületű operációs rendszerről lévén szó, a felhasználói tevékenységek kiválasztásának, megadásának alapvető eszköze valamilyen pozícionáló (kijelölő) eszköz (leggyakrabban az egér, illetve – elsősorban hordozható számítógépek esetében – az egér működésével azonos működési elvű eszköz), de mint azt a GUI alapelveinél is említettük, ez nem kizárólagos. Az egér használatakor a felhasználói tevékenység hatásával kapcsolatban a Windows az egér szimbólumának megváltoztatásával ad tájékoztatást, a legfontosabb állapotok a következők:

- nyíl  : alapértelmezett egérkurzor-alak, általánosan használható kijelölésre, kiválasztásra.
- kettős nyíl  : objektum méretezése.
- négyes nyíl  : objektum mozgatása.
- homokóra  : (az előzőleg kiválasztott) tevékenység végrehajtásának időtartama alatt megjelenő kurzor, amennyiben a végrehajtás alatt más tevékenység nem végezhető.
- tiltás  : a kijelölt művelet (az egér jelenlegi helyén) nem végezhető el, nem értelmezett(hető).
- a különböző alkalmazásokban a fentiek mellett különböző egéralakok is elképzelhetők, pl. szövegbevitelkor a szövegkurzor: , stb.

Az egér kezelésével kapcsolatban a Windows rendszerszinten két gombot különböztet meg, alapértelmezés szerint a bal oldali az elsődleges (az egyes tevékenységek kiválasztásához használandó), a jobb gomb az ún. gyors (vagy helyi) menük (olyan dinamikusan változó parancsokból felépülő menük, amelyek aktuális tartalma az egér pillanatnyi helyzetétől függ) megjelenítését teszi lehetővé. (A két gomb szerepe felcserélhető, továbbá az adott eszköz hardverkiépítésétől függően további szolgáltatások is definiálhatóak: pl. harmadik gombhoz, görgő használatához, stb.) A gombok kezelése mellett a rendszerszinten definiált műveletek (események) az egyszeres (alapértelmezés szerinti hatása: kiválasztás) és a kétszeres (dupla, alapértelmezés szerinti hatása: végrehajtás) kattintás, valamint a vonszolás (az egér elsődleges gombjának nyomva tartása mellett az egér mozgatása, alapértelmezés szerinti végrehajtása: mozgatás).

A mutatóeszköz használata mellett az alapvető tevékenységek billentyűzetről is megadhatók, rendszerint valamilyen billentyű-kombináció (egy vagy több vezérlőbillentyű lenyomva tartása közben kell a megfelelő billentyűt lenyomni) formájában. A teljesség igénye nélkül néhány általánosan használható billentyűkombináció:

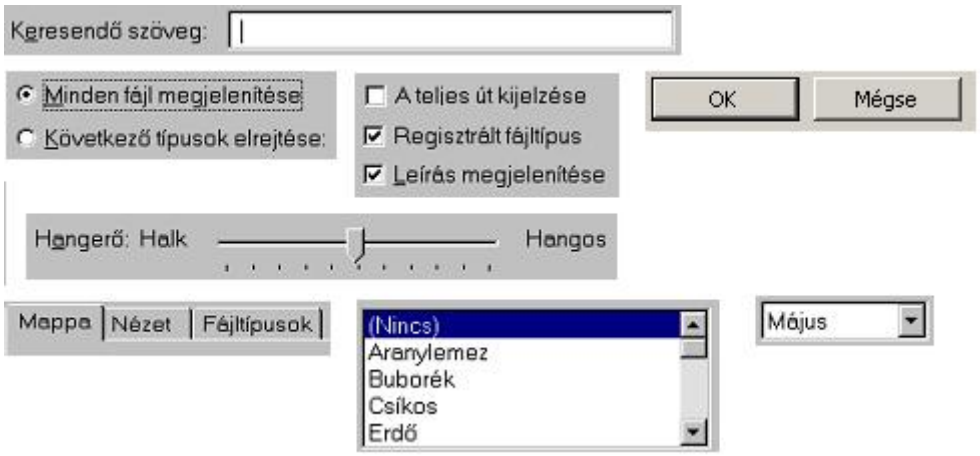
- CTRL+Esc – START menü aktiválása
- F1 – súgó (az aktív ablakra vonatkozóan)
- F3 – keresés (az állománykezelő alkalmazásokban)
- F2 – objektum átnevezése (az állománykezelő alkalmazásokban)
- SHIFT+Del – objektum fizikai törlése (az állománykezelő alkalmazásokban)
- Alt+Esc – következő ablak aktiválása
- Alt+F4 – aktív ablak bezárása
- F10 – menüsor elérése

- Alt+<betű> - a <betű> kiemeléssel (menü szövegében aláhúzott karakter) jelölt menüelem kiválasztása
- SHIFT+F10 – gyorsmenü
- Alt+Enter – kiválasztott objektum tulajdonságai
- CTRL+C, CTRL+X, CTRL+V – vágólap kezelése (másolás, kivágás, beillesztés)
- ⊞+M – minden ablak minimalizálása (ikonállapotban a Tálcára)
- ⊞+D – Asztal megjelenítése
- PrtScr („Print Screen") – az Asztal aktuális kinézeti képének másolása a Vágólapra (Az Alt vezérlőbillentyűvel együtt használva csak az aktuális ablak tartalmát másolja.)

(A fenti felsorolás természetesen nem teljes, a ⊞ szimbólummal jelölt vezérlő billentyű pedig nem található meg minden billentyűzeten.)

A felhasználói munkavégzés során a tevékenységek természetesen nem csak az Asztalra (vagy annak részeire), illetve az ablakok szabványos elemére vonatkozhatnak: a különböző felhasználói információk megadására számos további (grafikus) eszköz is rendelkezésre áll, ezeket nevezzük vezérlőknek. A vezérlők jelentősége túlmutat az operációs rendszerek kezelésének témakörén, hiszen bármilyen alkalmazásban a felhasználói beavatkozás lehetőségét határozzák meg – ebben az értelemben akár programozási komponenseknek is felfoghatóak –, azonban mivel működésük erősen kötődik az operációs rendszer használatához, ezért a legfontosabbakat az alábbiakban (3-14. ábra) összefoglaljuk:

- szöveg- (vagy beviteli) mező: billentyűzetről származó adat fogadását és megjelenítését végző vezérlő.
- rádiógomb: egyszeres kiválasztást lehetővé tevő vezérlő: általában több, egymást kizáró alternatívát jelenít meg, az egyik kiválasztásának hatására az (esetleges) korábbi kiválasztás törlődik.
- jelölőnégyzet: többszörös kiválasztást lehetővé tevő vezérlő: a rádiógombtól annyiban különbözik, hogy az alternatívák között megengedi több kiválasztását is.
- nyomógomb: kiválasztó vezérlő, általában tevékenységhez kötődik.
- toló- („pot-") méter: kiválasztás típusú vezérlő, értéktartományban történő dinamikus (analóg) kiválasztást tesz lehetővé.
- lap (fül): felsorolás típusú vezérlő, tartalma további vezérlők tematikus gyűjteménye, használatával egyetlen párbeszéd-ablakon belül több vezérlőcsoport is megadható
- lista: felsorolás típusú vezérlő, előre definiált értékek közötti választást tesz lehetővé.
- kombinált lista: felsorolás típusú vezérlő, az előre definiált értékek kiválasztása mellett (a listában nem szereplő) új érték megadását is lehetővé teszi (tulajdonképpen egy szövegmező és egy lista együtt)



3-21. ábra: Párbeszédablakok vezérlői

Az eddig megismert eszközöket számos kernel-szolgáltatás egészíti ki, amelyek közül külön említést érdemel a Vágólap. A Vágólap az operatív memóriának egy olyan része, amely valamennyi alkalmazás számára elérhető (emlékezzünk: szegmentált memória-kezelés esetén ilyen memória-terület létezése egyáltalán nem triviális!). A Vágólapra vonatkozó műveleteket az adatmozgatás iránya és jellege szerint csoportosíthatjuk: másolásnál a kijelölt objektum marad a helyén, tartalmáról képződik egy másolat a Vágólapon, kivágásnál a kijelölt objektum eredeti helyéről törlődik, és csak a Vágólapon érhető el a továbbiakban, beillesztés során pedig a Vágólap aktuális tartalma a kurzor aktuális helyére másolódik. A Vágólap kezelésére vonatkozó parancsok (nevük az egyes műveletek nevei) elérhetők minden alkalmazás Szerkesztés menüjében, a legtöbb alkalmazás eszköztárában, a helyi menükben és akár billentyű-kombinációk formájában is. A Vágólap kezelésével kapcsolatban két dolgot kell megjegyeznünk: az egyik, hogy alapértelmezés szerint vágólapból csak egy van (egyes alkalmazások lehetővé teszik több vágólap kezelését is – pontosabban ebben az esetben is csak egyetlen közös területről van szó, de az alapértelmezéstől eltérően ezen a területen több, különböző forrásból származó információ is tárolható), így minden, a vágólapra adatot mozgató művelet (másolás, kivágás) törli az aktuális tartalmat (a beillesztés viszont nem! – tehát a vágólap aktuális tartalma többször is felhasználható); a másik, hogy a vágólap tartalma csak olyan alkalmazásban használható fel, ami képes az aktuális típusú információ fogadására (pl. egy képet elhelyezve a vágólapon, ha megpróbáljuk egy formázatlan szöveget tartalmazó dokumentumba beilleszteni – ld. Jegyzettömb –, nem járunk sikerrel).

3.2.4. Állománykezelés

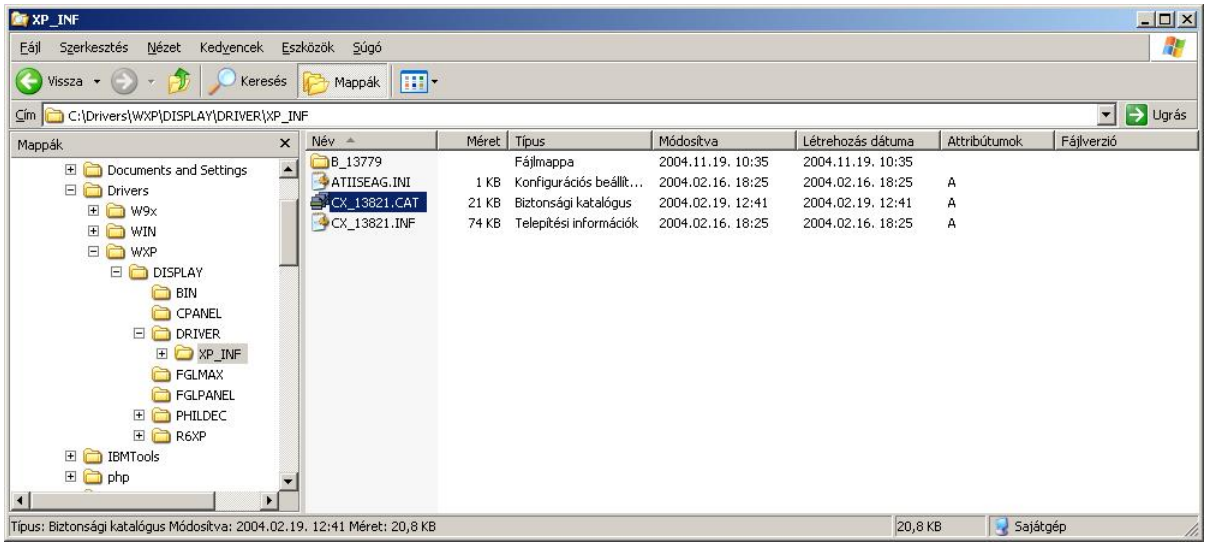
A grafikus felületű operációs rendszerekben az ablakkezelő mellett van még egy szabványosnak tekinthető komponens, az operációs rendszer állománykezelő szolgáltatásainak grafikus megfelelője. (Ez többé-kevésbé természetes is, ha belegondolunk a GUI működési elvébe: az ablakkezelő eseményvezérlést végez, de csak a GUI komponensek vonatkozásában – márpedig ahogy azt a Windows példájában láthattuk, ezen komponensek között szerepel(het)nek olyanok is: mappa, állomány, parancsikon, amelyek nem a grafikus felületnek, hanem az állomány-rendszernek képezik a részét.) A Windows ezt az elemet Explorer-nek (Intézőnek) hívja. A Windows Intéző tehát egyrészt valamennyi, az állományszervezéssel kapcsolatos művelet megvalósításáért felelős eleme a grafikus környezetnek, másrészt a Windows ugyanígy nevezi (nem véletlenül!) a grafikus felület (rendszerszintű) állománykezelő komponensét is. (A Sajátgép és az Intéző ugyanaz a folyamat, csak eltérő alapértelmezett felületi beállításokkal.)

Ahogy azt az állománykezelő rendszerek elméleti áttekintése során láttuk, az operációs rendszer ezen komponensének feladatai az állományok logikai és fizikai

elhelyezéséhez kötődnek (a felhasználó szempontjából természetesen a logikai szint az elsődleges). Az Intéző tehát az állományszervezés logikai szintjén megjelenő feladatok elvégzésének eszköze.

A Windows Intéző munkaterülete alapvetően két részre oszlik: a bal oldali a „böngészősáv”, a jobb oldali a „megjelenítési terület”: az előbbi a tevékenységek, az utóbbi az objektumok (mappa, állomány) kiválasztására szolgál (mindkét rész megjelenési módja változtatható a Nézet menü parancsainak segítségével). Az Intézőben elvégezhető tevékenységek mind a logikai állományszervezés struktúrájához (meghajtóra (kötetre), könyvtárra (mappára) és állományokra vonatkozó műveletek), mind hagyományos tevékenységeihez (létrehozás, módosítás (jellemzők, elhelyezkedési információk megváltoztatása), törlés) illeszkednek. Az egyes műveletek többnyire mindhárom megismert vezérlési mód (a menürendszer, az eszköztár ikonjai vagy a gyorsmenük használata) segítségével elvégezhetők.

A következőkben ezen műveletek közül (az alapvető állománykezelési tevékenységek elvégzésének lépésenkénti ismertetésétől eltekintve) azokat tekintjük át, amelyek a Windows XP-ben a megszokottól eltérő vagy azokat kibővítő szolgáltatásokat nyújtanak.



3-22. ábra: A Windows Intéző felülete

Fájlrendszerek: a Windows XP alapvetően mindhárom (a Microsoft esetében hagyományosnak mondható) fájlrendszert képes kezelni: FAT(16), FAT32, NTFS – de hatékony, és az operációs rendszer szolgáltatásait maximálisan kihasználó módon csak NTFS partíción. (FAT alapú partíciók használata csak abban az esetben indokolt, ha az adott partíció tartalmának a Windows korábbi verzióit futtató számítógépeken is elérhetőnek kell lennie, a FAT alapú partíciók adatvesztés nélkül konvertálhatók NTFS partícióvá – ez visszafelé nem érvényes!) Az egyes fájlrendszerek között komoly különbségek húzódnak meg mind az elérhető szolgáltatások körében, mind a technikai korlátok mértékében:

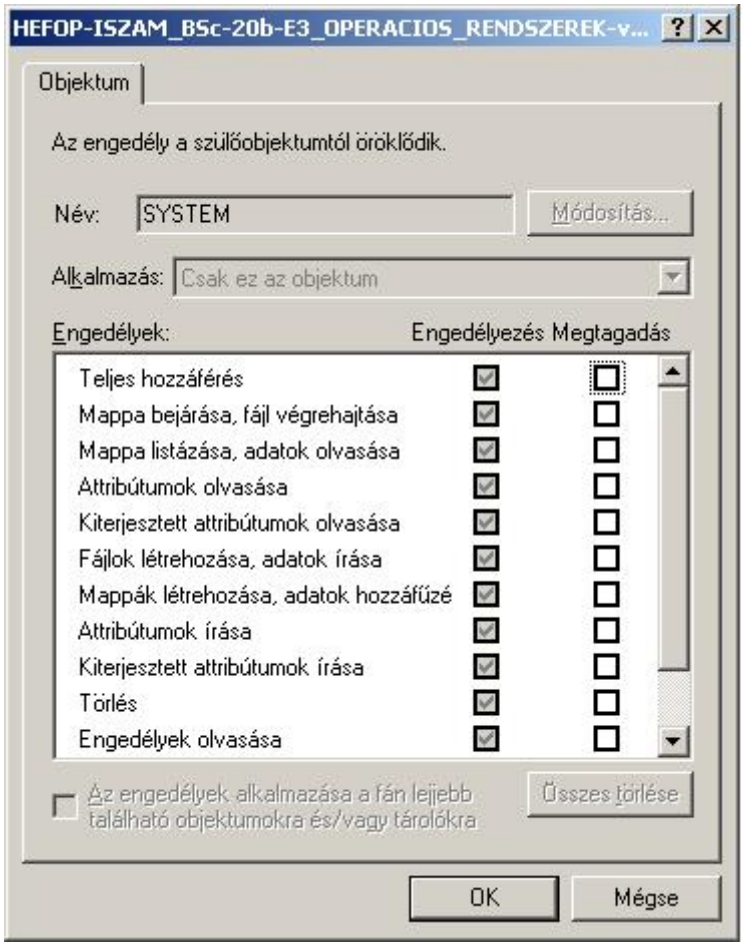
3-3. Táblázat: A Windows XP által támogatott fájlrendszerek

	FAT	FAT32	NTFS
Kötetméret	1 MB – 4 GB	512 MB – 2 TB (elvi) 512 MB – 32 GB (WXP gyak)	10 MB -
Fájlméret	max. 2 GB	max. 4 GB	nem korlátozott
Hosszú fájlnevek	alkalmazható	alkalmazható	alkalmazható

Jogosultság-kezelés	nincs	nincs	alkalmazható
Tömörítés	nincs	nincs	alkalmazható
Titkosítás	nincs	nincs	alkalmazható
Kvóta	nincs	nincs	alkalmazható

Az NTFS működési újdonságai közül az egyik legfontosabb a dinamikus kötetek kezelése. Dinamikus kötetet dinamikus lemezen lehet létrehozni – ez utóbbit pedig a lemezkezelés beépülő modul segítségével a hagyományos háttértár konvertálásával készíthetjük el. A dinamikus lemezek lehetővé teszik több lemezes kötetek kialakítását, a tárterület dinamikus növelését, illetve RAID rendszerekre jellemző szolgáltatások (csíkozás, tükrözés) megvalósítását.

Az NTFS az állománytulajdonságok kibővült körét tartja nyilván az egyes háttértárakon elhelyezkedő információkról, ezek közül felhasználói szempontból a jogosultsági rendszerhez kapcsolódó állománytulajdonságok a legfontosabbak. Tekintettel arra, hogy a Windows XP alapvetően több-felhasználós operációs rendszer, természetes, hogy az egyes állományokra vonatkozóan a felhasználók (vagy felhasználói csoportok) szintjén meghatározható az elvégezhető műveletek köre. Az NTFS partíción elhelyezkedő állományok esetében a jogosultsági rendszer kétszintű: alapvetően összetett jogok beállítását támogatja (ezek: írás, olvasás (és végrehajtás), módosítás), de elemi jogok segítségével ennél lényegesen finomabb műveleti szabályozás is elképzelhető. (Az elemi jogok beállításához az adott objektum tulajdonságai között a Biztonság fülön található Speciális parancsgomb segítségével juthatunk.)

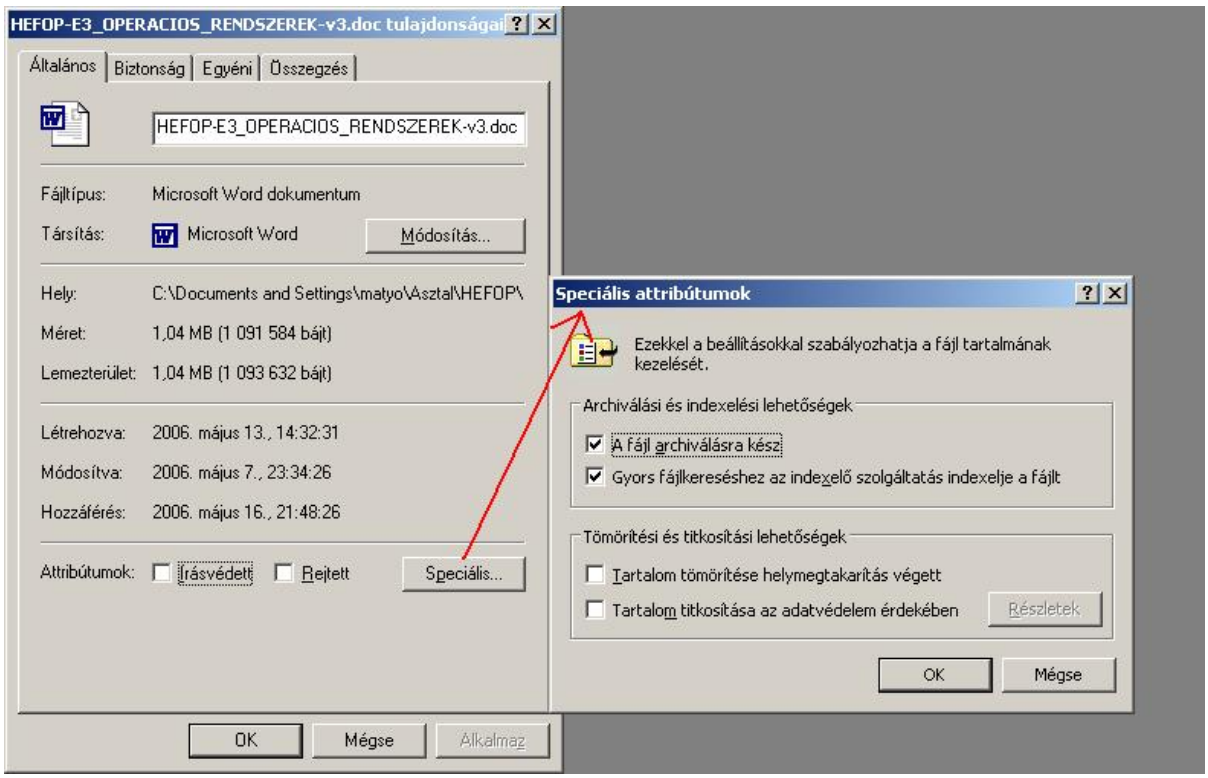


3-23. ábra: elemi állományszintű jogosultsági beállítás

A jogosultsági rendszerhez szorosan kötődik a szintén csak NTFS partíción elhelyezkedő állományok esetében alkalmazható titkosítási művelet. A titkosítás felhasználói szempontból az állomány egyik (kiterjesztett) attribútuma, az operációs rendszer EFS (Encryption File System) szolgáltatása valósítja meg. (Kiterjesztett attribútumok alatt a Windows hagyományos RHSA („read only” – írásvédett, „hidden” – rejtett, „system” – rendszerfájl, „archiv” – archív(ált) állomány) állomány-jellemzőit kiegészítő attribútumokat értjük. A Windows XP kezeli (már csak kompatibilitási okokból is) a négy hagyományos attribútumot – annak ellenére, hogy grafikus felületen (az állomány tulajdonságai között) csak kettőt, az írásvédelmet és a rejtettséget jeleníti meg.) A titkosítást az állomány (vagy mappa) tulajdonságai között az „Általános” fül „Speciális” parancsgombja segítségével állíthatjuk.

Amennyiben egy állomány rendelkezik „titkosított” attribútummal, úgy a háttértáron nem az eredeti tartalma, hanem annak egy kódolt változata kerül letárolásra. Ennek ellenére az állomány a hagyományos állományokkal azonos módon használható (a titkosítás-visszafejtés „transzparens” folyamat) – legalábbis a titkosítást végző felhasználó számára, bárki más „hozzáférés megtagadva” hibajelzést kap. A titkosítás attribútuma vonatkozhat mappára is, ebben az esetben (értelemszerűen) a mappa teljes tartalma titkosításra kerül.

(Technikailag a titkosítás során a felhasználó egyedi azonosítójával (Secure ID, SID) mint kulccsal történik a titkosítás. Mivel az SID a felhasználó létrehozásakor olyan módon generálódik, ami biztosítja az egyediségét, ezért ha letöröljük a titkosító felhasználót, hiába hozunk létre ugyanolyan jellemzőkkel egy újat, az előző felhasználó által titkosított állományok nem lesznek hozzáférhetőek…)



3-24. ábra: kiterjesztett attribútumok használata

Szintén a kiterjesztett attribútum-készletbe tartozik a röp-tömörítés tulajdonság is. A Windows XP alapvetően két tömörítési eljárással rendelkezik: a „tömörített mappák” szolgáltatás felhasználói tevékenység, a röp-tömörítés (a titkosításhoz hasonlóan a felhasználó számára észrevehetetlen – transzparens – módon megvalósuló) rendszerszolgáltatás. A röp-tömörítés a titkosítással azonos lapon kapcsolható attribútum, amennyiben egy állomány (vagy mappa) rendelkezik ezzel az attribútummal, úgy a tartalma a háttértárra történő írási

művelet során automatikusan tömörítésre kerül (megnyitáskor pedig – értelemszerűen – kitömörítésre).

A „tömörített mappák” ezzel szemben nem automatikus tömörítési eljárás (lényegében egy ZIP kódolású tömörítő algoritmus megvalósításáról van szó, amelyet integráltak az operációs rendszer szolgáltatásai közé), hanem felhasználói tevékenységről: amennyiben egy állományt (vagy mappát) be szeretnénk tömöríteni, úgy a (Fájl menü vagy a gyorsmenü) „Küldés” parancs „Tömörített mappák” elemét kiválasztva a tömörítés megtörténik. A tömörített mappa tulajdonképpen nem más (a hagyományos tömörítő programok: ZIP, RAR, ICE, stb. terminológiájával), mint egy archívum – amelyet azonban a Windows XP a mappákkal analóg módon kezel. Ennek megfelelően ebben a tömörítési módszerben is tetten érhető bizonyos mértékű transzparens működés: amennyiben már létezik egy ilyen mappánk, úgy állományokat helyezve ebbe a mappába megtörténik egy betömörítési művelet, állományokat kimozgatva ebből a mappából pedig lezajlik egy kibontási folyamat.

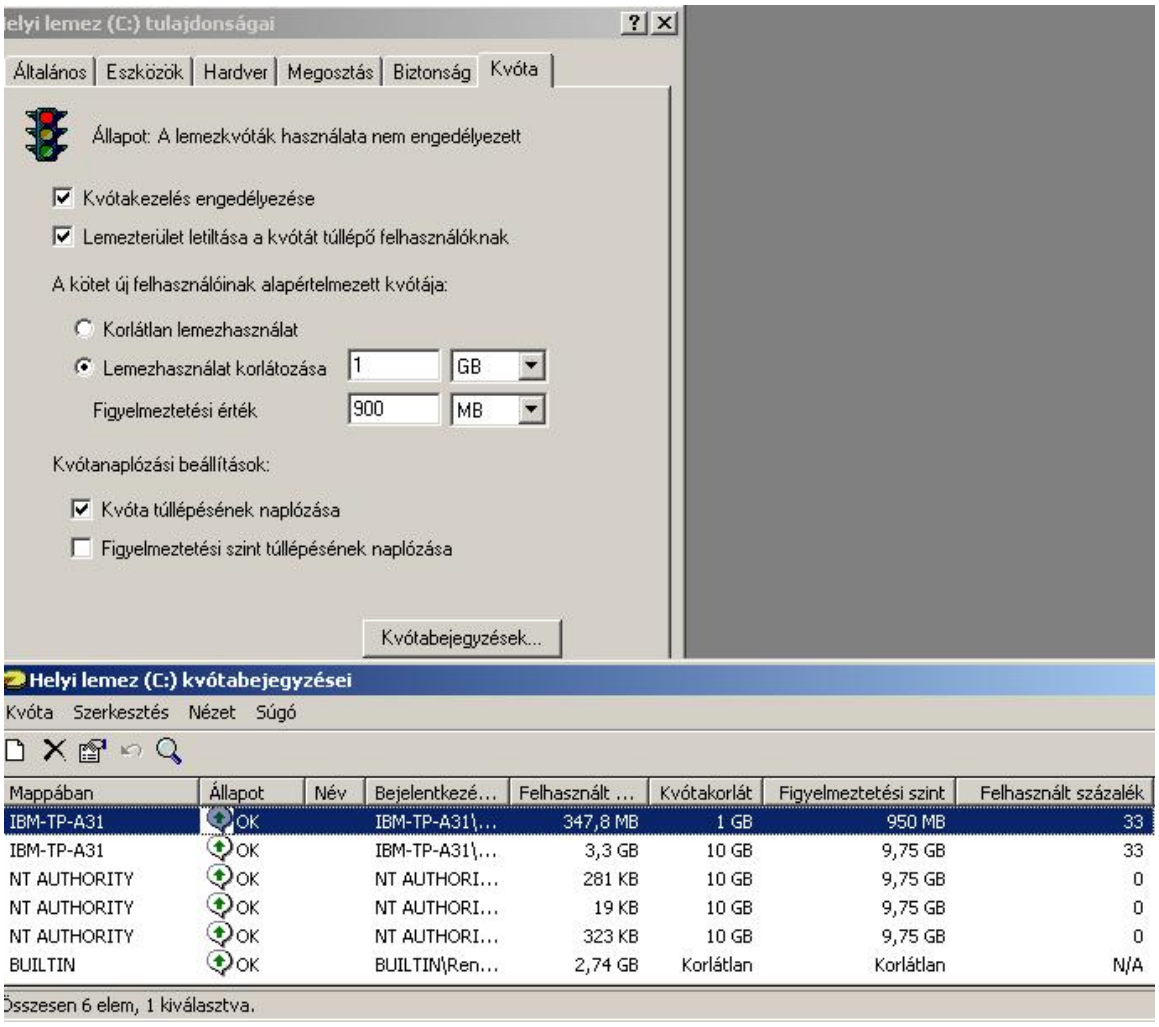
A két módszer azonban (a felhasználási terület jellemzőinek megfelelően) nem azonos módon működik: a röp-tömörítés gyorsabb, ugyanakkor (a tömörített állapot méretét tekintve) kevésbé hatékony, mint a tömörített mappák. (Ez természetes, ha belegondolunk abba, hogy röp-tömörítés használatakor a felhasználó nem elsődlegesen a tömörítés műveletével akar foglalkozni, hanem az állomány tartalmával, ezzel szemben a tömörített mappák használatakor maga a tömörítés a cél.)

(Végezetül egy utolsó megjegyzés a kiterjesztett attribútumok működésével kapcsolatban. Az ilyen attribútummal rendelkező állományok helyváltoztatása időnként (első ránézésre) meglepő eredményeket produkál:

1. ha lemásolunk egy röp-tömörített állományt egy ilyen attribútummal nem rendelkező könyvtárba, úgy törlődik ez az attribútum (azaz a célhelyen már nem lesz tömörített),
2. ha ugyanezt az állományt mozgatással helyezzük át az előzővel azonos könyvtárba, úgy a tömörítettsége megmarad.

A magyarázat az állománykezelő rendszer működésében rejlik: a mozgatás az állománykezelő rendszer számára csak a katalógus-bejegyzés módosítását jelenti, tehát (az operációs rendszer szerint legalábbis) ebben az esetben az állomány helye nem változik meg…)

Ahogy az a fentiekből is kiderült, a Windows XP az állománykezelés hagyományos feladatait számos többlet-szolgáltatással egészíti ki, amelyek ha úgy tetszik, kényelmi eszközök, de ha szem előtt tartjuk a több-felhasználós működési modellt, akkor ezek szükségszerű komponensek. A helyfoglalás korlátozása is egy ilyen eszköz, amely a hálózati operációs rendszerek jellemző szolgáltatásai közül került a Windows XP eszköztárába. A kvóták használatát kötetszinten lehet engedélyezni (a kötet tulajdonságai között található „Kvóta” fül segítségével). Amennyiben egy kötet rendelkezik kvótával, úgy az egyes felhasználók legfeljebb a kvótában megadott területet foglalhatnak le az adott köteten – beállítás kérdése, hogy a kvótában beállított adatmennyiség elérése után további adatmentés az adott felhasználó számára nem lesz lehetséges, vagy csak egy bejegyzés készül a rendszernaplóba a túllépésről.



3-25. ábra: Lemezkvóták beállítása és lekérdezése

Az állománykezelés gyakorlatában a megszokott műveletekhez képesti eltérések ismertetésekor a típusok szerepéről kell még szót ejtenünk. A Windows XP a hagyományos állománytulajdonságok (azonosító, méret, időbélyegek, jellemzők) közül az azonosító tulajdonság végének külön jelentőséget tulajdonít. Szokás ezt kiterjesztésnek nevezni, más források típusként hivatkoznak rá, technikailag az állomány azonosítójában az utolsó pont után elhelyezkedő karakterekről van szó. A típus jelentősége abban áll, hogy az operációs rendszer ez alapján meghatározza az adott állománnyal elvégezhető (nem állomány-, hanem alkalmazás szintű) műveletek körét, illetve hozzárendeli (amennyiben elérhető az adott számítógépen) az állományt feldolgozni képes alkalmazás(oka)t – ezt nevezzük társításnak. A Windows XP-ben lehetőség van egy típushoz többszörös társítási leírást is létrehozni, ilyen módon ugyanazon állománnyal (a felhasználás célja függvényében) más és más alkalmazások is képesek dolgozni. Ez az oka annak, hogy amennyiben megváltoztatjuk egy állomány azonosítóját, akkor az operációs rendszer egy (a kezdő felhasználót gyakran megtévesztő és elbizonytalanító) üzenetben figyelmeztet a típusváltásból adódó (lehetséges) társítási problémákra.

A társítás, illetve az állomány-típusok kezelése azonban közel sem triviális: alapértelmezett telepítés mellett azok a típusok, amelyeket az operációs rendszer „ismer” – azaz rendelkezik az adott típusra vonatkozó társítási listával –, az Intézőben nem jelennek meg. Ezt az „Eszközök” menüpont „Mappa beállításai” parancsának „Nézet” fülén lehet felülbírálni („Ismert fájltípus kiterjesztésének elrejtése”), és ugyanezen párbeszédablak „Fájltípusok” füle szolgál a társítások áttekintésére, módosítására. (Magát a társítást

egyébként közvetlen módon is elvégezhetjük a „Fájl” menüpont vagy a gyorsmenü „Társítás” parancsának használatával – ugyanez a parancs szolgál a többszörös társítással rendelkező típusok esetében az aktuálisan használni kívánt alkalmazás kiválasztására is.)

Természetesen a társítás művelete csak akkor alkalmazható eredményesen, ha az adott típushoz hozzárendelt alkalmazás képes feldolgozni az állomány tartalmát: egyrészt amennyiben olyan típussal találkozunk, amelynek a kezeléséhez nem rendelkezünk megfelelő programmal, azt társítva sem fogjuk tudni (érdemben) megnyitni, másrészt attól, hogy egy állományhoz másik alkalmazást társítok, attól annak a tartalma még nem fog megváltozni (ha pl. a JPG típushoz hozzárendeljük a Jegyzettömböt, attól a grafikus állományból nem lesz szöveg…)

3.2.5. Segédprogramok, kommunikációs és egyéb alkalmazások

Hagyományosan az operációs rendszerekkel szemben a felhasználónak „mindössze” annyi elvárása volt, hogy biztosítsa számára a számítógép kezeléséhez szükséges felületet. Ha visszagondolunk az X-Window alapfilozófiájára, a grafikus felületű operációs rendszerekben is mindössze egy ablakkezelő (és egy állománykezelő) áll rendelkezésre (elvileg). Manapság azonban az operációs rendszerek számos olyan szolgáltatást is tartalmaznak, amely nem elsődlegesen a rendszer működését, sokkal inkább a felhasználó kényelmét vagy megelégedését hivatottak szolgálni. Nem kivétel ez alól a Windows XP sem, a következőkben néhány általános célú rendszerkomponenssel (vagy ha így jobban tetszik – hiszen tulajdonképpen így kell(ene) neveznünk ezeket az alkalmazásokat – segédprogrammal) ismerkedünk meg.

Alapértelmezett telepítés mellett a START menüben a Kellékek csoport tartalmazza a Windows XP ilyen elemeit.

Szövegszerkesztést támogató alkalmazásból alapértelmezés szerint kettő is van a Kellékek között: a Jegyzettömb formázatlan szöveg kezelését lehetővé tevő szerkesztő program (az angol nyelvben a különböző típusú, szöveget feldolgozó alkalmazásokra – annak funkcionalitása szerint – különböző (kifejező) elnevezések, ezt a csoportot „text editor”-nak hívják), míg a WordPad egy alapszintű formázási műveleteket kezelni képes szövegszerkesztő („word processor”). Ennek megfelelően a Jegyzettömb elsődlegesen olyan jellegű szöveges tartalmak létrehozásának, szerkesztésének vagy megjelenítésének az eszköze, amelyek formai jellemzőket nem tartalmaznak – ilyenek pl. a különböző programkódok. A WordPad segítségével pedig egyszerű formázott szöveget (pl. egy hivatalos iratot, levelet) készíthetünk el. A Számológép a nevéhez mérten sok meglepetéssel nem szolgál, de azt mindenképpen érdemes róla tudni, hogy két működési módja van: normál nézetben hagyományos négy alapműveletes számoló eszköz, azonban tudományos nézetben trigonometriai, statisztikai (és az informatikában jelentős számrendszereket is támogató) műveletek elvégzésére is alkalmas – a két mód között a Nézet menü segítségével lehet váltani.

Szintén a kellékek között érhető el a karakteres felület megjelenítésére szolgáló Parancssor, valamint a Windows Intéző is.

A Windows XP nagy hangsúlyt fektet felhasználói igények minél teljesebb kiszolgálására, ennek következménye a hálózati műveleteket és a multimédia alkalmazását (mint a felhasználói szempontból két legfontosabb informatikai alkalmazási terület) támogató segédprogramok minden korábbinál bővebb választéka. A sok vitát kiváltó Internet Explorer továbbra is az operációs rendszer alapértelmezett böngészőprogramja maradt (bár immár opcionálisan eltávolítható), az Outlook Express (a Microsoft Office programcsomag részét képező Outlook nevű komplex csoportmunka-támogató program „leegyszerűsített” változata) az elektronikus levelezést, a Messenger a közvetlen (on-line) kommunikációt („csevegés”), a

NetMeeting a hálózaton keresztüli adat- és információ-megosztást teszi lehetővé (ez utóbbi alkalmazás egyben a multimédia támogatására is példa, hiszen az állományokon túl audió és videó tartalmat is képes a hálózat felhasználói között továbbítani, pl. videó-konferencia formájában), végül pedig a „Távoli asztal" szolgáltatás segítségével terminál-szolgáltatás használatára (azaz fizikailag egy másik számítógépről használni egy távoli számítógép erőforrásait) nyújt lehetőséget az operációs rendszer.

Ami a multimédia támogatását illeti, a Paint nevű grafikus szerkesztő (rajz-) program segítségével rasztergrafikus képek készíthetők, szerkeszthetők (a PrtScr billentyűvel lementett képernyőképek pl. kiválóan alakíthatók illetve menthetők a használatával – jelen jegyzet ábráinak egy része is ilyen módon készült), a Médialejátszó egyaránt alkalmas audió- és videófájlok lejátszására, a Hangrögzítő illetve a Movie Maker pedig az ilyen formátumú állományok létrehozására, szerkesztésére.

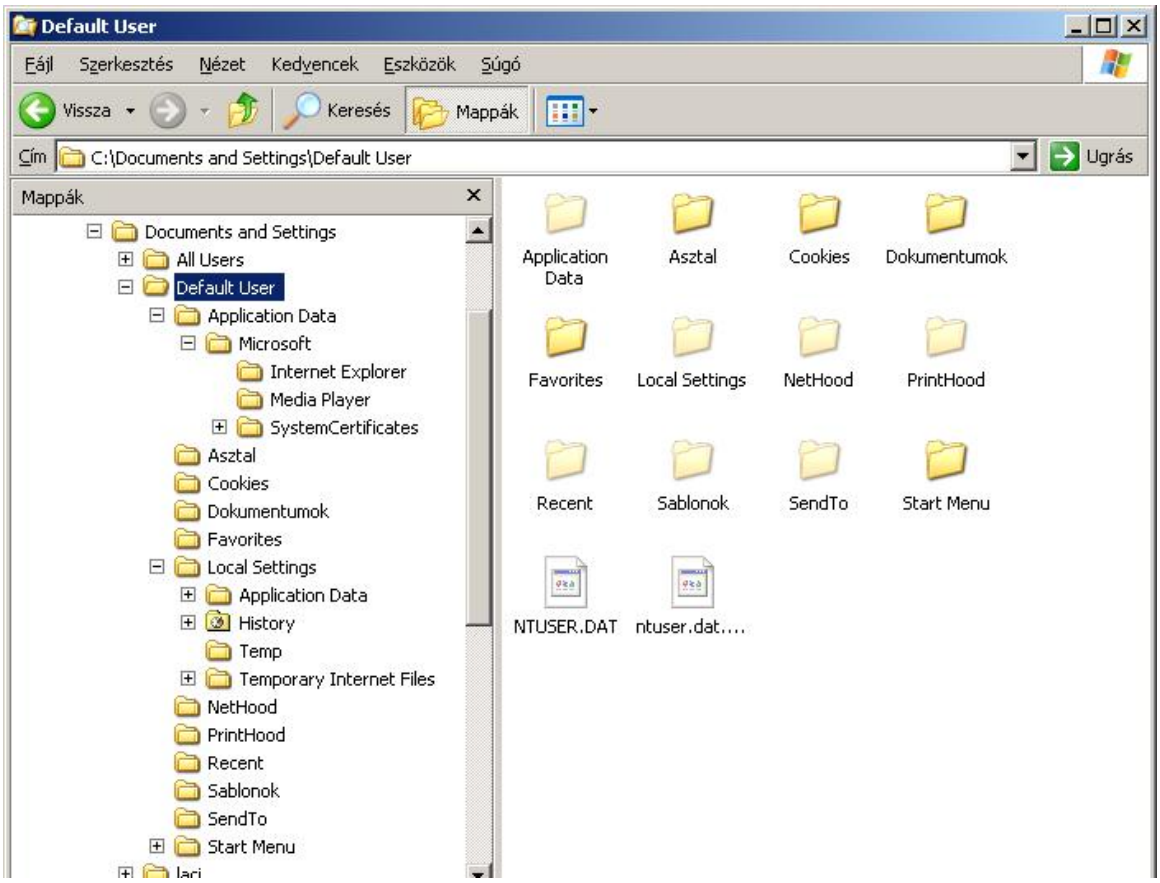
3.2.6. Felügyeleti eszközök

A segédprogramok között külön csoportot képeznek – nem feltétlen a megvalósító alkalmazások megjelenése vagy elhelyezkedése szempontjából, sokkal inkább a nyújtott szolgáltatások tekintetében – azok az alkalmazások, amelyek az operációs rendszer működésének megváltoztatására szolgálnak, ezek a felügyeleti eszközök. (Ezeket az alkalmazásokat, szolgáltatásokat általában alapértelmezett felhasználói jogosultság mellett nem vagy csak korlátozott funkcionalitással használhatjuk – a teljes körű felügyelethez kitüntetett felhasználói jogosultságok (a Windows XP fogalmai szerint „kiemelt felhasználó" vagy „rendszergazda") szükségesek.)

3.2.6.1. Vezérlőpult

A felhasználó számára (is) elérhető felügyeleti eszközök közül a Vezérlőpult a legismertebb és a legáltalánosabban használható. Alapértelmezés szerint elérhető a START menü Beállítások csoportján keresztül, de megjelenik az Intézőben is (a Sajátgép – ami egy, háttértárak logikai szervezési egységei (kötetek) fölött elhelyezkedő virtuális „gyűjtő", leginkább egy „szuper-gyökérkönyvtár"-nak fogható fel – alatt). A Vezérlőpult segítségével lényegében az egyes felhasználók saját felhasználói felületet alakíthatnak ki: beállítható az Asztal megjelenése (Megjelenítés), a kijelölő és vezérlő eszközök működése (Billentyűzet, Egér), az operációs rendszer környezete (Dátum és idő, Területi és nyelvi beállítások, Biztonsági központ, Energia-gazdálkodás, Hálózati kapcsolatok), az állománykezelő alapértelmezései (Mappa beállítások), a nyomtatási szolgáltatások (Nyomtatók és faxok), a telepített alkalmazások (Programok telepítése és törlése) stb. (Az egyes beállítási eszközök működésének részletes ismertetésére terjedelmi okokból nem térünk ki, de javasoljuk kipróbálásukat.)

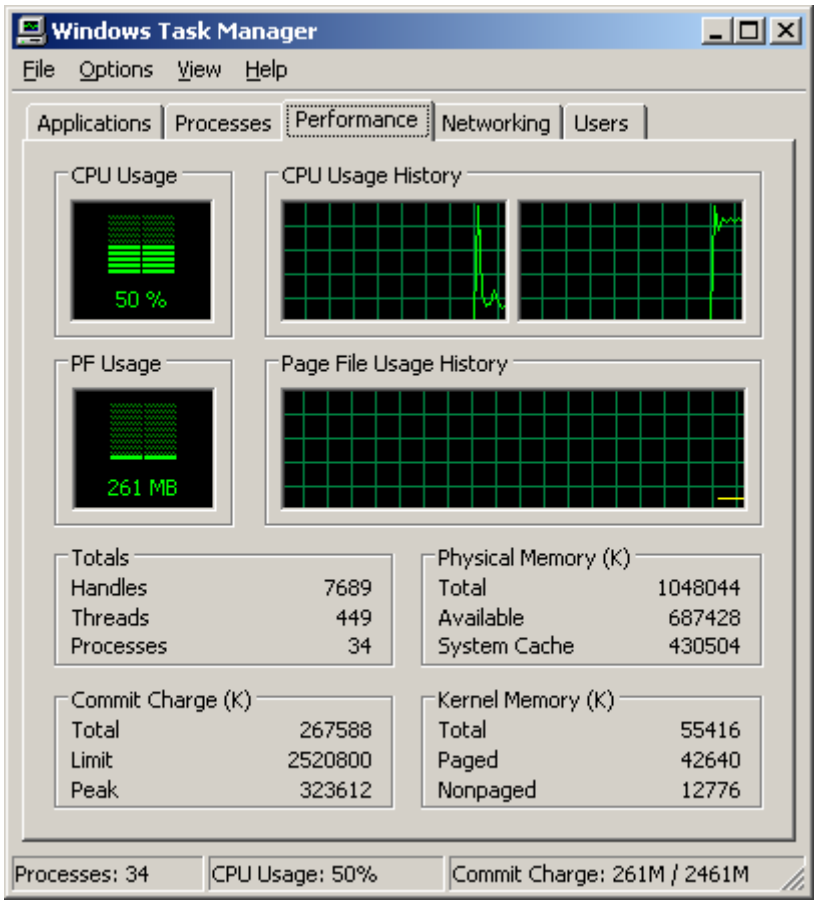
A felhasználói felület kialakításával kapcsolatban meg kell említenünk a felhasználói profilokat. A profil az egyes felhasználók saját (személyre szabott) beállításainak gyűjteménye – a Windows XP filozófiája szerint egy könyvtárstruktúra. Alapértelmezés szerint a rendszerkötet gyökérkönyvtárában található „Documents and settings" mappában minden felhasználói névhez tartozik egy (a felhasználói névvel azonos nevű) mappa, amely (jobbára rejtett mappák formájában) tárolja a felhasználó környezeti (és bizonyos műveleti) beállításait. (Ez a mappa csak az adott felhasználó számára hozzáférhető (ld. állományszintű jogosultsági rendszer) – ez a tény adja a jelentőségét az „All users" nevű speciális profilnak, amelyben azok a beállítások tárolódnak, amelyek valamennyi felhasználóra egységesen vonatkoznak – ilyenek pl. a START menü mindenki számára elérhető elemei, valamint a „Megosztott dokumentumok" mappa.)



3-26. ábra: profilok

3.2.6.2. Feladatkezelő

Az operációs rendszerek általános alapfeladatai között említettük a monitoring (a rendszerben működő folyamatok áttekintésére, szükség szerint beavatkozásra lehetőséget adó) szolgáltatásokat – a Windows XP esetében ez a komponens a Feladatkezelő. Elindítható a Tálca gyorsmenüjéből vagy a CTRL-ALT-DEL (hagyományosan a számítógép újraindítására szolgáló) billentyű-kombináció segítségével. Használatával áttekinthető a rendszerben futó alkalmazások és az alkalmazások által használt folyamatok állapota, erőforrás-használata, valamint a rendszer általános teljesítményére vonatkozó mérőszámok és a hálózati felhasználásra vonatkozó adatok. Igazi jelentőségét azonban az áttekintésen túl a beavatkozás lehetősége adja: segítségével elindíthatunk és leállíthatunk alkalmazásokat (pl. a nem válaszoló (lefagyott) vagy a nem kívánt (vírus) programokat).



3-27. ábra: Feladat-kezelő

3.2.6.3. Registry

A rendszerleíró adatbázis (közismert – és a magyar megfelelőnél lényegesen rövidebb – angol elnevezésével a registry) a Windows XP „lelke”. Lényegében egy hierarchikus szerkezetű adatbázis, amelyben valamennyi, a rendszer működésével kapcsolatos változó vagy változtatható értékkel rendelkező paraméter tárolódik (lényegében a korábbi verziók konfigurációs fájljainak: pl. CONFIG.SYS, AUTOEXEC.BAT, WIN.INI, SYSTEM.INI, stb. egységes szerkezetbe integrált változata).

A rendszerleíró adatbázis kezelése a következő esetekben történik:

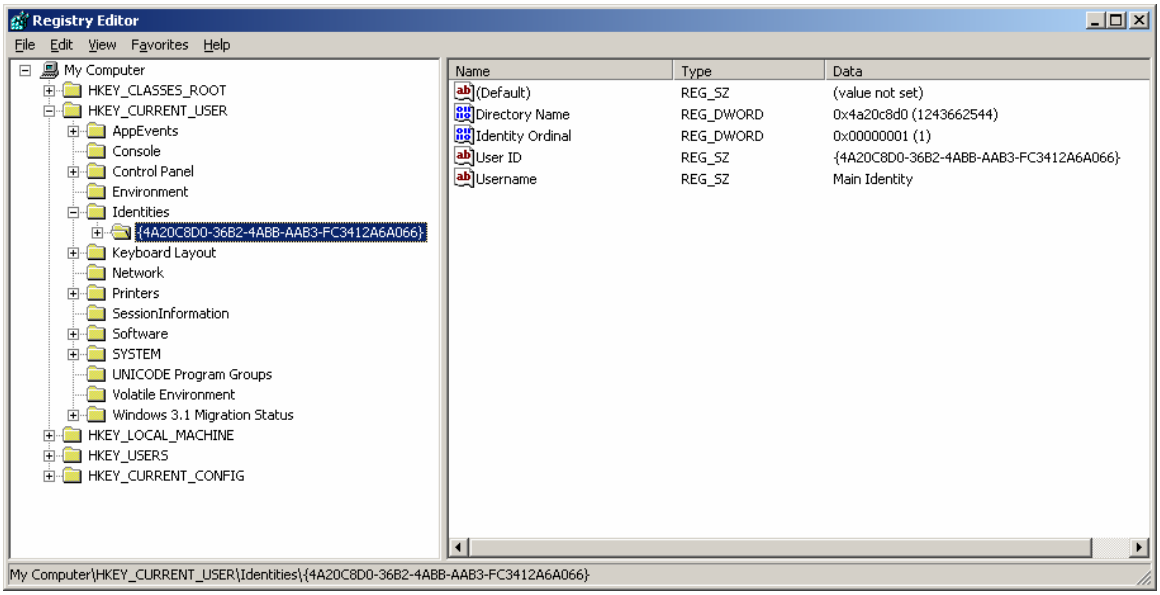
- rendszerindításkor a számítógép hardverével kapcsolatos (detektált) beállítások ide íródnak,
- a kernel indulásakor innen veszi a hardverinformációt (és az egyéb indítási paramétereket), majd saját adatait beírja,
- az eszközmeghajtók innen veszik a működésükhöz szükséges paramétereket (induláskor), és ide írnak vissza (módosítás, telepítés során),
- egyes programok telepítőprogramjai ide írják a program beállításait,
- a registry tartalmazza a teljes biztonsági adatbázist is.

Ez azt jelenti, hogy gyakorlatilag minden, a rendszerbeállításokat megmutató és megváltoztató program (Vezérlőpult, MMC-k, stb.) a rendszerleíró adatbázist írja és olvassa.

A registry szerkezete hasonlóan a háttértárak könyvtár-szerkezetéhez, de az elnevezések különböznek: a teljes struktúra (az egész adatbázis) neve „fa”, az egyes elágazások a „részfák”, az egyes részfa-sorozatok végén pedig az az elem, ami a nevén kívül

valamilyen konkrét értékkel is rendelkezik, a kulcs. Legmagasabb szintű részfából öt van (az egyes részfákra a részfa nevét alkotó szavak kezdőbetűiből képzett mozaikszóval szoktak hivatkozni):

- a **HKEY_LOCAL_MACHINE** a legfontosabb, valamennyi rendszerszintű beállítás (mind hardver, mind szoftver szinten) ebben a részfában található. A részfa legmagasabb szintű kulcsai speciális kulcsok: a **HARDWARE** a számítógép hardverének konfigurációs beállításai (az itt levő adatok a rendszer indításakor dinamikusan jönnek létre, a lemezen levő registry adatbázisban nem szerepelnek), a **SAM** és **SECURITY** a biztonsági adatbázis és a biztonsági rendszer beállításai (soha ne módosítsuk a Registry Editorral!), a **SOFTWARE** a számítógépen telepített programok (beleértve az operációs rendszert is) környezeti beállításai (a **HKEY_CURRENT_USER** részfában is van egy **SOFTWARE** kulcs, ez azonban az egyes programok felhasználónként külön szabályozható, egyéni beállításait tartalmazza).
- **HKEY_USERS**, és a **HKEY_CURRENT USER** a felhasználók beállításainak gyűjteménye (az utóbbi az éppen dolgozó felhasználóra vonatkozik), lényegében a felhasználói profil (pontosabban annak egy része). A részfa második magas szintű kulcsa (S-...) alatt a felhasználó munkakörnyezetének beállításai vannak; a kulcs neve a felhasználó biztonsági azonosítószáma.
- **HKEY_CLASSES_ROOT** és a **HKEY_CURRENT_CONFIG** valójában a HKLM részfáinak másolatai, az elsőben az állománykezelésnél említett társítással kapcsolatos beállítások, a másodikban bizonyos paraméterek aktuális értéke (pl. nyomtatók beállításai) tárolódik.



3-28. ábra: Rendszerleíró adatbázis

A registry a regedit.exe (vagy a regedt32.exe) paranccsal indítható programmal szerkeszthető – ez azonban mindenképpen tapasztalt és a rendszer működésével, valamint a szerkeszteni kívánt kulcs hatásával tisztában levő felhasználót tételez fel. (Ahogy azt az előbb is említettük, a registryben a Windows működésének leírása található. Ez az adatbázis a rendszerindítás során részint írásra kerül (pl. hardverleltár), de túlnyomó részt kiolvasásra, és a kulcsokban foglalt értéknek megfelelően próbálja az operációs rendszer elvégezni a kijelölt műveletet – könnyű belátni, hogy amennyiben hibás értéket adunk egy kulcsnak, akkor az a következő indításkor (jobb esetben) figyelmen kívül marad, de (rosszabb esetben) akár

működésképtelenné teheti a rendszert. Éppen ezért a regedit lehetőséget biztosít a registry egészének vagy részének elmentésére, illetve szükség szerint visszaállítását (Fájl menü, Export… és Import… parancsa).

A registryben az egyes kulcsokat és változókat - az állományokhoz hasonlóan - elérési útjukkal adhatjuk meg, így például a számítógépünk nevét tartalmazó változó elérési útja a következő:

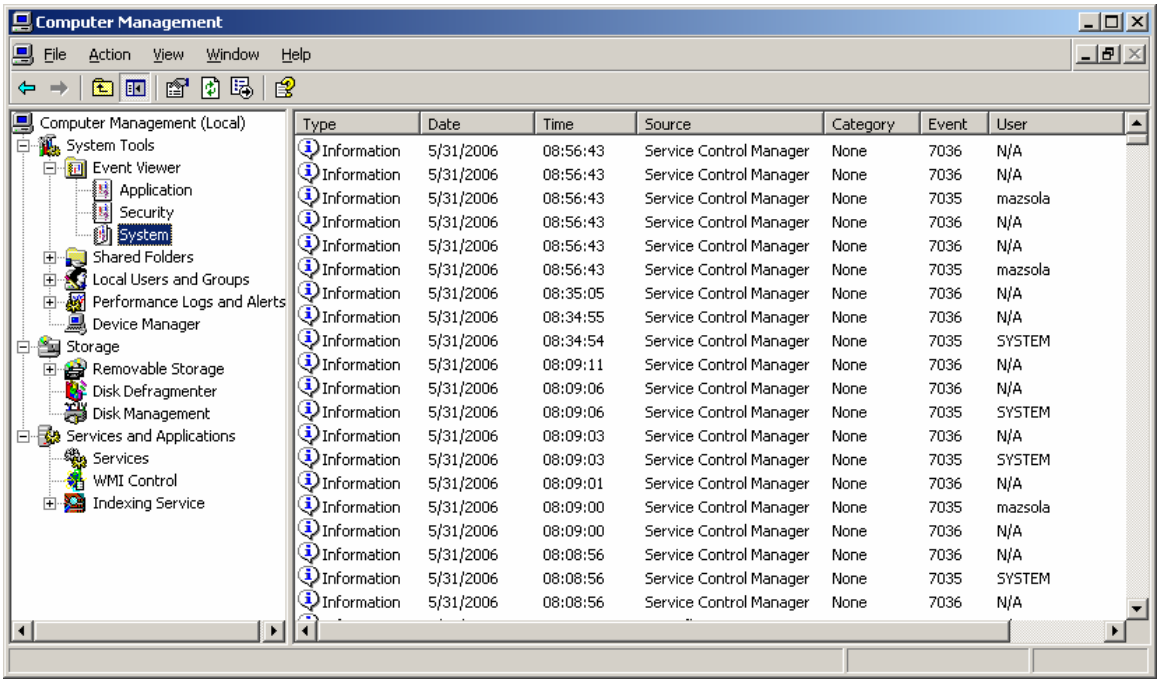
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\ComputerName\ComputerName

A registry jelentősége (a használatának a veszélyével együtt is) a teljes körű leíró képességében rejlik: minden beállítás elvégezhető a használatával – ilyen pl. a nem kívánt programtevékenység (a vírusok előszeretettel helyezik el saját indítási parancsukat a registry HKLM/Software/Microsoft/Windows/CurrentVersion/Run kulcsok valamelyikébe) kezelése, de ide tartoznak azok a rendszerbeállítások is, amelyekhez nem tartozik Vezérlőpultbeli vagy egyéb felügyeleti eszköz.

3.2.6.4. MMC

A Microsoft Management Consol („Felügyeleti Konzol”) a felügyeleti eszközök újabb generációjának egyik képviselője, és (mint a Windows XP esetében oly sokszor már) a hálózati környezetből került át az asztali operációs rendszerek világába. Lényegében maga az MMC egy üres keretrendszer, amelybe alkalmas módon megírt felügyeleti eszközök (ezek a konzol beépülő moduljai) felhasználásával tetszés szerinti felügyeleti tevékenységtár kialakítható.

A Vezérlőpult egy része (hardver-kezelő, felügyeleti eszközök) is (előre definiált) MMC – legszembetűnőbb – és legösszetettebb – alkalmazására példa a Sajátgép gyorsmenüjének Kezelés eleme (3-22. ábra).



3-29. ábra: Előre definiált felügyeleti konzol

Saját felügyeleti konzol létrehozásához először az MMC-t kell elindítanunk (a START menü Futtatás eleme segítségével, a parancs „mmc”), majd a konzol Fájl menüjének „Beépülő modulok hozzáadása” parancsával vehetjük fel a kívánt modulokat. Hálózati környezetben az MMC alapértelmezés szerint nem csak a saját számítógép komponenseinek felügyeleti

eszköze, hanem valamennyi, a hálózatba (ügyfélként) kapcsolt számítógép beállítása elvégezhető a segítségével, ezért az egyes modulok esetében (ahol értelmezhető), megadható, hogy a modul a saját gép, vagy egy távoli gép erőforrásait jelenítse meg, illetve kezelje!

3.2.6.5. Scriptek

A grafikus felület fentebb említett eszközei ugyan majdnem teljeskörű felügyeleti eszközrendszert biztosítanak, de a Windows XP továbbra is támogatja a parancsmódú rendszerfelügyeleti szolgáltatásokat is. Ez egyrészt megnyilvánul a már említett helyreállítási konzol alkalmazásában – emlékeztetőül: ebben az esetben az operációs rendszer állományrendszerének fájl-szintű kezelésére van lehetőségünk karakteres felületen, parancsok segítségével –, másrészt megnyilvánul a különböző programozott felügyeleti és ellenőrzési eszközök futtatásának képességében. Ez utóbbi alatt azt kell értenünk, hogy a Windows XP tartalmaz egy natív kódfordítót (WSH: Windows Scripting Host), ami (korlátozott) Visual Basic (VBScript) illetve Java (Jscript) kódban írt programok futtatására képes. Ezekkel az eszközökkel objektum szinten az operációs rendszer teljes erőforrás-készlete elérhető és menedzselhető – azaz megfelelő mélységű rendszerismeret és programozói tudás birtokában szinte bármilyen tevékenység megírható, amit aztán az operációs rendszer mint saját komponenst képes futtatni. (A filozófia alapvetően a rendszerfelügyeletre, illetve a rendszer hangolására – pl. az operációs rendszer felhasználó által hiányolt szolgáltatásokkal történő kiegészítésére – lehetőséget biztosító eszközökként definiálja a rendszer programozhatóságának támogatását, de mint oly sok esetben, az „élni és visszaélni” között itt is csak hajszálvékony a különbség: a pár évvel ezelőtt hatalmas „sikert” aratott „IloveYou” vírus is egy VBScript volt...)

Példaként álljon itt egy (a Microsoft tudásbázisának VBScript példákat tartalmazó oldaláról származó) egy egyszerű VBScript kód, ami a fájlrendszerben egy fájl tulajdonságát kezeli:

```
Dim Signer, File, ShowUI, FileOK
Set Signer =CreateObject("Scripting.Signer")
File = "c:\newfile.wsf"
ShowUI = True
FileOK = Signer.VerifyFile(File, ShowUI)
If FileOK Then
WScript.Echo File & " is trusted."
Else
WScript.Echo File & " is NOT trusted."
End If
```

3-30. ábra: VBScipt (minta)

3.2.7. Telepítés

A Windows alapú rendszerek telepítése olyan, mint a labdarúgás: aki csinálta már, annak megvan a saját elképzelése, hogy miért jó, ahogy csinálta – aki még nem, az meg tudja, hogy hogyan tudná jobban csinálni az előbbinél. A Windows XP telepítése egy „varázsló” által vezérelt telepítési folyamat, ami a konkrét rendszer kialakításába csak minimális beavatkozási lehetőséget biztosít a felhasználónak. Ez persze nem azt jelenti, hogy minden Windows XP egyforma, csupán azt, hogy a telepítés során egyéni jellemzők megadására a telepítőkészlet nem biztosít lehetőséget – ez (a Microsoft filozófiája szerint egyrészt szükségtelen, másrészt ha mégis szükséges, akkor) a telepítést követő utólagos feladat.

Dacára annak, hogy minden felhasználónak, minden rendszermérnöknek megvan a saját elképzelése (és nem ritkán bevált telepítési módszere) a rendszer telepítésével (vagy adott esetben újratelepítésével kapcsolatban), a következőkben bemutatjuk a (Microsoft által

javasolt) telepítés menetét. Hangsúlyozzuk, hogy ez a nem egy utasítás, csupán ajánlás, amelyet mindenki a saját igényei és lehetőségei ismeretében módosíthat!

A telepítés első lépése a **tervezés**. Ebben a fázisban célszerű megismerkedni a telepíteni kívánt operációs rendszer általános jellemzőivel, de mindenképpen tisztában kell lenni a telepíteni kívánt rendszer legfontosabb általános hardver és hálózati paramétereivel. A telepítés során a Windows automatikus hardverfelismeréssel dolgozik (saját eszközmeghajtók, illesztőprogramok hozzáadására, cseréjére nincs lehetőség), így célszerű a telepítés előtt tanulmányozni a Microsoft **HCL** (*Hardver Compatibility List, kompatibilis eszközök jegyzéke*) ajánlását – amennyiben olyan eszközzel rendelkezünk, amely ebben a listában nem szerepel, az a telepítés során gondot okozhat (jó esetben ismeretlen eszközként nem vesz róla tudomást a rendszer, de szélsőséges esetben a telepítés sikertelenségét is okozhatja, ezért az ilyen eszközöket célszerű a telepítés során eltávolítani (vagy letiltani), és csak a rendszer sikeres telepítése után hozzáadni).

Külön említést érdemel a rendszer *teljes újratelepítése* (pl. rendszerösszeomlás – e tekintetben Windows XP rendelkezik automatikus helyreállítási eszközökkel – vagy verzió-frissítés esetén). Ebben az esetben az alábbi szempontok figyelembe vétele javasolt:

- készítsünk biztonsági mentést (ha lehetséges) a rendszer jelenlegi állapotáról,
- szüntessük meg az esetlegesen alkalmazott RAID szolgáltatásokat (tükrözés...),
- ha működő rendszert szeretnénk frissíteni, állítsuk le a nem rendszerszintű (vagy a telepítéshez nélkülözhető) szolgáltatásokat (vírusirtók, hálózati szolgáltatások, stb.),
- (a Microsoft ajánlása szerint a rendszerfrissítés időtartamára lehetőleg ne használjunk szünetmentes áramforrást...)

A telepítés során (ne feledjük: a Windows XP eredendően hálózati szolgáltatásokra tervezett operációs rendszer, feltételezi a hálózati kapcsolat meglétét!) szükségünk lesz még a hálózati azonosítással kapcsolatos információkra is (amennyiben több számítógépet tartalmazó rendszerben dolgozunk – pl. munkahely –, akkor a hálózati modellre (munkacsoport vagy tartomány) is, de a hálózati kapcsolat (alapértelmezés szerint TCP/IP) adataira mindenképpen).

A telepítés menete (az előbbiek szerint irányított módon) a következő lépésekből áll:

1. telepítés forrásának kiválasztása – alapértelmezés szerint a Windows XP a telepítőkészletet tartalmazó CD-ROM-ról indítva települ (adott esetben ehhez a számítógép BIOS beállításainak módosítása szükséges!), de elképzelhető hálózati erőforrás (megosztott mappa) is, mint forrás
2. hardver-ellenőrzés
3. minimális kernel betöltése – általános (többé-kevésbé szabványos) eszközvezérlők, alapszintű (karakteres felületű) rendszerkonzol
4. EULA (licenc-szerződés)
5. elérhető (korábbi) verziók ellenőrzése, telepítés módjának meghatározása (frissítés – helyreállítás – újratelepítés)
6. tárolási rendszer kialakítása (a Windows XP alapértelmezés szerint egy boot- és egy rendszer-partíciót kezel (a kettő lehet fizikailag azonos), de célszerű – amennyiben a háttértár mérete lehetővé teszi – további partíciók kialakítása (egy a rendszernek, ennek minimális mérete 1,5 GB, ajánlott mérete ennek a duplája, és

legalább egy az alakalmazások, illetve adatok számára – ilyen módon (is) csökkenthető a hibás működésből adódó adatvesztés veszélye)

7. rendszerállományok másolása
8. hardver-elemek konfigurálása – PnP kompatibilis illetve a HCL-ben szereplő eszközök esetében
9. egyéni beállítások (regionális jellemzők, tulajdonos adatai, termékkulcs, stb.)
10. hálózati beállítások
11. telepítendő komponensek másolása
12. újraindítás – bejelentkezés – kész!

A telepítés előbb vázolt sémáján túl elképzelhetők más telepítési megoldások is (központosított telepítés, ahol több – azonos paraméterekkel rendelkező – számítógép telepítése végezhető egy parancsköteg segítségével, illetve a lemezkép-másolás (a módszert támogató legismertebb alkalmazás neve után „ghost-olás”-ként emlegetett módszer).

Ellenőrző kérdések

1. Melyek a Windows alapú operációs rendszerek általános jellemzői?
2. Hasonlítsa össze a fontosabb Windows verziókat!
3. Hogyan érhető el és milyen feladatok elvégzésére alkalmas (adjon példákat) a Windows karakteres felülete?
4. Ismertesse a Windows grafikus felületének szabványos felépítését, az egyes elemek jellemzőit és szerepét!
5. Ismertesse a rendszerindítás és a rendszer-leállítással kapcsolatos alapvető tevékenységeket, lehetőségeket!
6. Melyek a Windows XP alapértelmezett felhasználói felületen elérhető fontosabb (rendszer)szolgáltatásai?
7. Hasonlítsa össze (alkalmazhatóság szempontjából) a Windows XP által támogatott fájlrendszereket!
8. Mutassa be azokat a fájlrendszerhez kapcsolódó szolgáltatásokat, amelyekkel az NTFS rendelkezik!
9. Hasonlítsa össze a röptömörítés és a tömörített mappák szolgáltatásokat (működés, felhasználás szempontok szerint)!
10. Ismertesse a Windows XP állományszintű jogosultsági rendszerének elemeit, működését!
11. Hasonlítsa össze (alkalmazhatóság, elvégezhető műveletek köre, stb. szempontok alapján) a megismert felügyeleti eszközöket!
12. Ismertesse a felügyeleti konzol kezelésével kapcsolatos alapvető műveleteket!
13. Hogyan épül fel a rendszerleíró adatbázis?

Ábrajegyzék

Táblázatok jegyzéke

Irodalomjegyzék

1. Olajos Zs. – Magó Zs.: Operációs rendszerek (Számalk, 2003)
2. Magó Zs. – Nagy S.: Hálózati felhasználói ismeretek (Számalk, 2003)
3. Knapp G. – Operációs rendszerek (LSI, 1998)
4. Kis B. – Lovassy Zs.: Windows Server 2003 (SZAK Kiadó, 2005)

Internetes források (2006. májusi állapot szerint):

5. WikiPedia: en.wikipedia.org (angol), hu.wikipedia.org (magyar)
6. Operációs rendszerek áttekintése (angol): www.csee.wvu.edu/~jdm/classes/cs258/OScat
7. Operációs rendszerek összehasonlítása (angol): www.osdata.com
8. Linux: http://www.linux.org/

Ellenőrző kérdések

A kérdésre kattintva megkapja a választ

1. fejezet

1.1. Melyek a harmadik generációs számítógép működésében megjelent legfontosabb technológia újítások?
1.2. Hogyan valósítható meg a korszerű operációs rendszerekben a feladat-megosztás?
1.3. Hogyan csoportosíthatjuk az operációs rendszereket?
1.4. Hasonlítsa össze a karakteres és a grafikus felületű operációs rendszerekben a parancskiadás módját!
1.5. Ismertesse az időosztásos és a valós idejű elveken működő operációs rendszerekben a folyamatok időzítésének módjait!

2. fejezet

2.1. Adjon meg legalább 3 különböző definíciót az operációs rendszer fogalmára!
2.2. Sorolja fel az operációs rendszerek alapfeladatait!
2.3. Ismertesse az operációs rendszerek rétegmodelljét!
2.4. Milyen rendszerhívás-típusokat különböztethetünk meg?
2.5. Ismertesse a megszakítások kezelésének általános menetét!
2.6. Melyek a legfontosabb folyamat-állapotok?
2.7. Mire szolgálnak az ütemezési stratégiák?
2.8. Mit jelent a „rögzített címzés” és az „átlapoló technika” (a memóriakezelés vonatkozásában)?
2.9. Miben a virtuális memóriakezelés alapelve?
2.10. Parancsmódú rendszerek esetén hogyan néz ki egy utasítás általános szerkezete?
2.11. Mi az „X11”?
2.12. Hogyan csoportosíthatók a GUI „ablak” és „ikon” típusú objektumai?
2.13. Mit jelent a „röptömörítés”?
2.14. Mit értünk (mágneslemezes háttértárak esetében) „klaszter” alatt?
2.15. Magyarázza meg a következő fogalmakat: állomány, könyvtár, kötet!
2.16. Melyek az operációs rendszerek által leggyakrabban használt állomány-tulajdonságok?
2.17. Milyen műveletek végezhetők el a könyvtár típusú bejegyzések írása során?
2.18. Mit értünk „abszolút elérési út(vonal)” alatt?
2.19. Mi a „fájl elhelyezkedési táblázat” (FAT)?

3. fejezet

3.1. Mi a Linux?
3.2. Melyek a Unix (Linux) rendszerek legfontosabb tulajdonságai?
3.3. Milyen rétegekből épül fel egy UNIX alapú rendszer?
3.4. Miben különbözik az UID és a GID?
3.5. Mit jelent (UNIX alapú rendszerekben) a „démon (daemon)” kifejezés?
3.6. Mit tartalmaz Linux alatt a gyökér (/) fájlrendszer?
3.7. Mi a szerepe Linux alatt a(z) /usr fájlrendszernek?
3.8. Melyek az FHS által meghatározott fájl-kategóriák?
3.9. Melyek a Linux fájlrendszerben használt jogosultsági műveletei?
3.10. Miben tér el Linux alatt egy előtérben illetve háttérben futó folyamat egymástól?
3.11. Melyek a Linux leállítására szolgáló legfontosabb parancsok?
3.12. Soroljon fel 3-3, állományok illetve folyamatok kezelésére szolgáló Linux parancsot!
3.13. Mi a „K-panel” hés a „K menü”?
3.14. Hogyan (milyen forrásokból) telepíthető egy Linux rendszer?
3.15. Melyek a Linux alapvető partíciói?
3.16. Melyek a Linux alapú rendszerekben alkalmazott csomag-szabványok?

3.17. Miben áll a „linuxconf” program jelentősége?
3.18. Melyek Linux alatt a felhasználó létrehozásához szükséges (kötelező) adatok?

4. fejezet

4.1. Melyek a Windows XP legfontosabb technológiai újításai?
4.2. Hogyan érhető el a Windows karakteres felülete?
4.3. Soroljon fel 3-3 Windows parancsot az állománykezeléshez, illetve a felhasználói környezet kialakításához kapcsolódóan!
4.4. Melyek a Windows grafikus felületének szabványos komponensei?
4.5. Miben különbözik a Windows készenléti és hibernált állapota?
4.6. Mi a felhasználói profil?
4.7. Melyek a Windows XP-ben alkalmazott összetett állományszintű jogok?
4.8. Hasonlítsa össze a Windows XP „röptömörítés” és „tömörített mappák” szolgáltatásait!
4.9. Mire szolgál a Windows XP „Feladatkezelő” eszköze?
4.10. Mi a „registry”?
4.11. Milyen lépéseken keresztül telepíthető a Windows XP?

Internetes linkek, hasznos weboldalak

Wikipedia, Internet-lexikon:

- http://en.wikipedia.org/wiki – kezdőlap (angol)
- http://hu.wikipedia.org/wiki – kezdőlap (magyar)

Áttekintés, gyűjtemények:

- http://operaciosrendszer.lap.hu (magyar)
- http://tunes.org/Review/OSes.html (angol)

Linux

- HUP Wiki: http://wiki.hup.hu
- Linux.hu: http://www.linux.hu
- Linux Online: http://www.linux.org (angol)
- Linux Bázis: http://www.linuxbazis.hu
- KDE: http://www.kde.org
- Gnome: http://www.gnome.org
- OpenOffice: http://www.openoffice.hu/index.php (magyar), http://www.openoffice.org (angol)
- Linux kézikönyv: http://www.szif.hu/~ahorvath/Sag/sag-hu/sag-hu.html

Windows

- hivatalos MS honlap: http://www.microsoft.com/en/us/default.aspx (angol), http://www.microsoft.com/hun (magyar)
- Windows operációs rendszerek: http://www.microsoft.com/hun/windows/default.mspx
- NetAkadémia Tudástár: https://www.netacademia.net/tudastar/
- Tippek, trükkök: http://lacy.hu/mrxp/index.php