Operációs rendszerek

Operációs rendszernek nevezzük azokat a program elemeket, melyek a felhasználói programok és a gép hardvere („Hardver erőforrások”) között helyezkednek el.

Egy számítógép működhet operációs rendszer nélkül is. Ez kisebb teljesítményű beágyazott rendszerekben szokásos: a teljes programot a felhasználó írja meg.

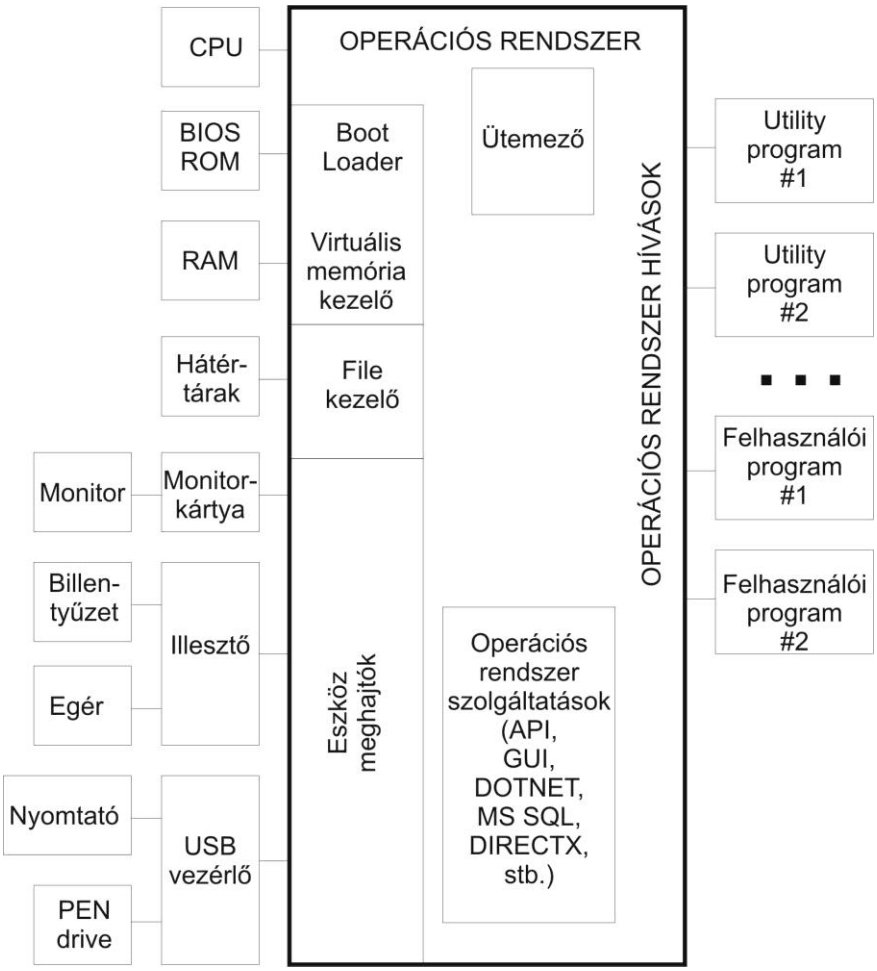
Az operációs rendszerek legfontosabb feladatai:

- a számítógép erőforrásainak (processzor, memória, diszk) kezelése
- a hardware elemek "elfedése" a felhasználó, ill. a programok elől, egységes, jól használható hozzáférési módszerek biztosítása
- az emberi felhasználó számára kényelmes, hatékony felhasználói felület kialakítása

Az operációs rendszer elemei:

- speciális operációs rendszer elem a PC BIOS
- eszköz meghajtók
- ütemező
- virtuális memória kezelő
- file kezelő
- felhasználói interface.

Az operációs rendszer felépítése:



Alapvető típusok:

- egyfeladatos - egyidőben csak egyetlen program fut
- többfeladatos (multitasking) - ugyanazon a processzoron több program is futhat egyszerre.
  Többfeladatos operációs rendszereknél többletfunkció: az erőforrások megosztása az egyes programok között, processzor ütemezés, memóriavédelem

- egyfelhasználós - csak egyetlen felhasználó használja
- többfelhasználós (multi-user) - több, egymástól független felhasználó kiszolgálására is felkészült.
  Többletfeladat: az egyes felhasználók adatainak/programjainak megvédése egymás elől, hozzáférési jogosultságok kezelése

(Sok szakkönyv csak a többfeladatos, többfelhasználós operációs rendszereket tekinti "igazi" operációs rendszernek.)

Tipikus példák:

- CP/M - 8-bites mikroszámítógépekhez (70-es évek vége). Leszármazottja:
- MS-DOS - IBM PC-hez. Ehhez egy grafikus kiegészítés volt a Windows 3.1.
- Mindkettő egyfeladatos, egyfelhasználós (Windows 3.1 próbálkozásokat tett a többfeladatosság felé, mérsékelt sikerrel).
  Továbbfejlesztések:
- Windows 95, Windows 98, Windows ME - többfeladatos (többé-kevésbé)
- Windows NT, Windows 2000 - más alapokon felépült, többfeladatos, többfelhasználós

- UNIX - 60-as évek legvégétől kezdődően, alapvetően többfeladatos, többfelhasználós; számos változata létezik, manapság a legelterjedtebb:
  Linux (PC-kre, 1991-ben készült, eredetileg diplomamunkának)

- MacOS - az Apple Macintosh számítógépek operációs rendszere. Többfeladatos, de alapvetően egyfelhasználós. Csak grafikus felhasználói felületet tartalmaz.

Még több száz további op. rendszer létezik, de ezek a legelterjedtebbek.

IBM PC BIOS

A PC-ben (és más univerzális számítógépekben) a programok RAM memóriában futnak, amelyik a gép bekapcsolásakor üres – pontosabban tartalma véletlenszerű. A PC-ben van egy nem nagy méretű ROM (EPROM, EEROM vagy FLASH), bekapcsoláskor az ebben lévő program – a BIOS (Basic Input Output System) – kezd el futni. A BIOS fő célja az, hogy valamilyen háttértárról – floppy disk, merev lemezes meghajtó, CD meghajtó - elkezdje betölteni az operációs rendszert, majd annak átadja a vezérlést. Ezt a folyamatot szokás Bootstrapping-nek (csizmahúzás) nevezni.

A BIOS-hoz tartozik a kisméretű telepről táplált kis kapacitású nem felejtő BIOS RAM (történelmi okokból CMOS RAM-nak is nevezik, mert valamikor csak ez a memória volt CMOS a PC-ben), amely az adott PC hardvernek az indításhoz szükséges paramétereit tartalmazza. Egyúttal egy Real Time órát is tartalmaz, amely kikapcsolt PC-nél is működik – ezért ismeri a PC a dátumot és időt bekapcsolás után.

A PC BIOS paramétereket a PC indulásakor, általában Ctrl-Alt-Del lenyomásával lehet megnézni, állítani.

Eszközmeghajtók

Az eszközmeghajtók vagy eszköz kezelők (Device driverek vagy Device handlerek) a különböző hardver eszközök felületét illesztik az operációs rendszer egységes felületéhez. Az eszközmeghajtók olyan programok, melyek az operációs rendszer oldaláról az eszközt működtető eljárás hívásokat tartalmaznak, másrészt kezelik a hardver megszakításokat. Ilyen hívások például „Nyomtass egy karaktert” vagy „Nyomtass egy sort”.

Ha a PC-hez új eszközt kapcsolunk, annak driver-ét is telepíteni kell. A telepítés lehetőségei:

- az operációs rendszer már tartalmazza az adott eszköz driverét, illetve az eszköz felülete szabványos. Ilyen a legtöbb merevlemezes meghajtóegység
- az eszközzel együtt kapott CD-n lévő programot kell futtatni az eszköz csatlakoztatása előtt (például nyomtatóknál)
- az eszköz csatlakoztatásakor az operációs rendszer automatikusan vagy félautomatikusan telepíti a drivert (például XP operációs rendszer pen-drive-nál)
- az internetről kell letölteni a drivert.

Az eszközmeghajtók jelentik az operációs rendszer alapszintjét, lehet, hogy csak ezt tartalmazza az operációs rendszer.

Tételezzük fel, hogy egy mikrokontrollert valamilyen feladatra akarunk alkalmazni. Az alkalmazáshoz „ki kell találni”, hogy a mikrokontroller erőforrásait pontosan hogyan kell használni, és megírni a kezelőprogramot. Ha ezt a program többi részétől elválasztva írjuk meg, máris kész az eszközkezelő, amelyet további alkalmazásokban is lehet használni.

Operációs rendszer szolgáltatások

API Application Programing Interface

GUI Graphic User Interface

DOTNET Microsoft .NET Framework (Program könyvtár általános célokra+virtuális gép program végrehajtásra)

MS SQL Microsoft SQL adatbázis kezelő program

DirectX Játék és Video interface

CPU kezelés: ütemezés

Egyfeladatos esetben nincs rá szükség, hiszen egyszerre csak egyetlen program fut. Többfeladatos operációs rendszerek esetén az ütemezés kulcsfontosságú funkció: az egyetlen processzort meg kell osztani a különböző programok között, hogy mindegyik futhasson.

Alapötlet: a tipikus programfutás során általában processzorigényes munkaszakaszok (pl. számítás végzése) és periféria műveletek (pl. merevlemez írás/olvasás) váltakoznak, tehát amíg az egyik program éppen a perifériaműveletre vár, addig a másik használhatja a processzort.

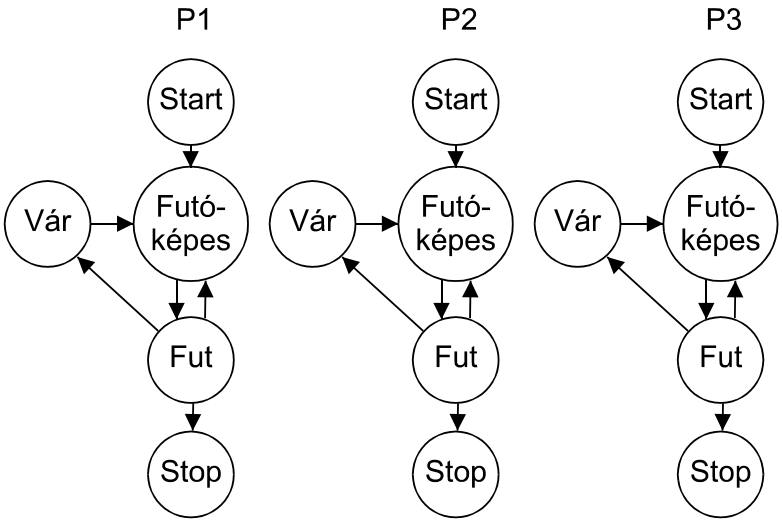
A modell a következő:

Több program van egyszerre a számítógép memóriájában. Ezek a következő állapotok valamelyikében lehetnek:

- Futóképes: ha megkapná a processzort, futna
- Fut: a futóképes folyamatok közül egyetlen kapja meg a processzort
- Vár: időzítés leteltére, vagy perifériára vár

A programot valamikor elindítjuk és leállítjuk – vannak olyan programok, melyek mindaddig futnak, amíg a számítógép „be van kapcsolva”.

Három program állapotait mutatja a következő ábra:



Egyszerre csak egy program fut. Az operációs rendszer dönti el, hogy a futóképes programok közül melyik fusson.
Az ütemezés két alaptípusa:

- kooperatív multitasking - az egyes programok saját maguk döntik el, hogy meddig futnak, ill. mikor mondanak le a futás jogáról más programok javára.

  Előnye: az operációs rendszert így jóval egyszerűbb kialakítani; az alkalmazások pontosan tudhatják, hogy mikor fog a működésük megszakadni
  Hátránya: csak akkor működik, ha az összes program betart bizonyos szabályokat, és véletlenül vagy szándékosan nem sajátítja ki a processzort (pl. programozási hiba miatt egy program végtelen ciklusba esik → az egész rendszer lefagy)

- preemptív multitasking - egy külső időzítő periodikusan megszakítja az éppen futó programot, és egy másikra kapcsol át

  Előnye: nem tud (olyan könnyen) "lefagyni" egy hibás vagy rosszindulatú programtól
  Hátránya: az op. rendszer bonyolultabb, a programokat úgy kell megírni, hogy a működésük bármikor megszakítható legyen.

Minden elterjedtebb többfeladatos operációs rendszer preemptív multitasking-ot alkalmaz (bár a Microsoft Windows verziók ütemezése nem egészen preemptív).

Az ütemezés másik fontos kérdése, hogy hogyan döntsük el, hogy amikor egy program futása megszakad, melyik másik program fusson tovább.

Példák ütemezési algoritmusokra:

- Fix prioritás: a programok fix sorrendben vannak, a sorban legelső futóképes kapja meg a processzort. A nagyobb prioritású programok nem foglalhatják túl hosszú ideig a processzort, mert akkor a kisebb prioritású program sohasem fut.
- SJF (Shortest Job First): az a program kapja meg a vezérlést, amelyik (várhatóan) a legkevesebb ideig fogja foglalni a processzort a következő perifériaműveletig
- round-robin - egyszerű körforgás: a programokat egy kör mentén rendezzük sorba, egy program leállásakor a körben utána lévő első futóképes program kapja meg a processzort. Garantáltan minden program sorra kerül.

- sorbanállás: amikor egy program futóképessé válik, beáll egy sor végére. A programfutás megszakadásakor a sorban első program kapja meg a processzort.
- vegyes: általában a fenti módszereket keverik: a sürgősebben végrehajtandó programoknak fixen nagyobb prioritása van, mint a többinek, egy csoporton belül pedig körforgás vagy sorbanállás működik. Megjegyezzük, hogy a megszakításos program prioritása feltétlenül nagyobb, mint az alapszinten futó programoké, hiszen a megszakítás kiszolgálás félbeszakítja az alapszinten éppen futó programot.

Memória kezelés

Az egyes programok számára ki kell jelölni a számítógép memóriájának egy darabját, amelyet csak az adott program használhat. Ez kétféleképpen történhet:

- statikusan - a program betöltésekor kap egy megadott méretű memóriaterületet, és azzal kénytelen megelégedni.
  - Előnye: egyszerű megoldani
  - Hátránya: ha egy programnak a futása közben több memória kellene, mint amennyit előre megkapott, nem tud még szerezni → valószínűleg nem tud tovább futni; ha viszont sokkal kevesebbet használ valójában, mint amennyit kapott, akkor más programoknak nem jut, tehát rossz a kihasználtság.

- dinamikusan - a programok az általuk használt memóriaterületet futás közben is növelhetik vagy csökkenthetik igény szerint.
- Előnye: minden program annyi és csak annyi memóriát használ, amennyire valóban szüksége van.
- Hátránya: ha sok program egymástól függetlenül foglal le és szabadít fel memóriaterületeket, akkor a szabad memóriaterület könnyen össze nem függő kis darabkákra töredezhet (tördelődés), amelyeknek összmérete akár elég is lenne egy újabb foglaláshoz, de mivel nem egybefüggő a szabad terület, nem lehet használni. Megoldások:
  - "ügyes" memóriafoglalás (a foglalandó terület méretétől függően más helyről foglaljuk le a következő blokkot; az egymással szomszédos felszabadult blokkokat egyesítjük)
  - szemétgyűjtés (garbage collection) - végigjárjuk a lefoglalt memória-darabokat, és "összetoljuk" őket, azaz az adatok megfelelő átmozgatásával eltüntetjük a köztük levő üres helyeket.
    Probléma: ehhez a memóriát foglaló programokat is megfelelően kell megírni; a szemétgyűjtési folyamat lassú, emiatt csak időnként lehet lefuttatni, amikor nem zavarja más programok működését
  - PC-ben a Windows által használt megoldás: program szegmentálás és virtuális memóriakezelés (ld. IA09-Személyi számítógép)
    Ehhez hardver támogatás kell !!!

Többfeladatos operációs rendszereknél felmerülő feladat a *memóriavédelem*: a programok csak az általuk birtokolt memóriaterületet írhatják/olvashatják, ill. csak az ott levő kódok futtathatják. Ezt is megoldja a PC-ben használt virtuális memóriakezelés: a futó program csak a neki kiosztott logikai szegmensekhez fér hozzá.
Ezt is kezeli a PC-ben alkalmazott virtuális memóriakezelés: a program csak a neki kiosztott logikai szegmenseket látja.

A memória elérési sebességének fizikai korlátai vannak. A ma széles körben elterjedt, nagy

mennyiségben is olcsó memóriachipek hozzáférési ideje (40-70 ns) már jóval nagyobb, mint a mai gyors processzorok ciklusideje (0.8-5 ns). A megoldást az ún. *cache memória* alkalmazása jelenti: kisebb méretű, gyorsabb (10-20 ns) memória elhelyezése a processzor és a fő memória között; esetleg még kisebb méretű, még gyorsabb (2-5 ns) memória beépítése magába a processzorba. Ez általában gyorsítja a memóriaelérést, mert a programok többsége a memóriát nem "összevissza" használja, hanem egy időben csak a memória egy kis részletével foglalkozik (lokalitás). Ha tehát egy memóriacímre való hivatkozáskor az illető cím közelében elhelyezkedő memóriaterületet egyszerre bemásoljuk a cache memóriába, a processzor a további műveleteket már nagy valószínűséggel a cache-ben fogja végrehajtani.

Lehetőség van arra is, hogy a programok a számítógépben ténylegesen jelen levő RAM-nál több memóriát érjenek el, oly módon, hogy a merevlemez egyes területeit - megfelelő kiegészítő hardware és az op. rendszer segítségével - hozzárendeljük a valójában nem létező memóriacímekhez, és az ilyen címekre való hivatkozáskor az adatokat a merevlemezről olvassuk, ill. oda írjuk (*virtuális memória*). Ily módon a programok számára elérhető memóriaterület gyakorlatilag korlátlan lesz, azonban a virtuális memória intenzív használatakor a rendszer nagyon lelassulhat (hiszen ugyanannyi adat átvitele a RAM-ból kb. 100000-szer gyorsabb, mint merevlemezről!), tehát a lemezműveleteket ennek megfelelően nagyon hatékonyan kell megszervezni.

Lemezkezelés

A mágneslemezre az adatokat blokkonként – úgynevezett clusterenként írjuk fel. A lemezen file-okat tárolunk. Egy file egy vagy több cluster-ben fér el. Megjegyezzük, hogy egy file clusterei nem szükségszerűen egymás után vannak a lemezen.
Az operációs rendszer legfontosabb feladata az ún. file-rendszer kialakítása, azaz a fix méretű blokkok (clusterek) halmazaként elérhető lemezen önálló, tetszőleges méretű - emberi felhasználó számára olvasható - névvel rendelkező adattároló egységek, azaz file-ok kialakítása. Ehhez a lemezen bizonyos mennyiségű területet a file-rendszer struktúrájának leírására használnak fel.

A ma használatos operációs rendszerek mindegyike az ún. hierarchikus fastruktúrát használja a file-rendszerek szervezéséhez (a file-ok könyvtárakba vannak csoportosítva, egy könyvtáron belül további könyvtárak is lehetnek). Ennek leírására alapvetően kétfajta adatszerkezetet kell kialakítani:

- foglaltsági információk: a lemez mely szektorai tartoznak létező file-okhoz és melyek szabadok. (Akár a foglalt, akár a még szabad szektorokat tárolhatjuk.)
- könyvtár- és file-információk: az egyes file-okat tartalmazó szektorok, a file-ok nevei, mérete, utolsó módosítás dátuma stb.

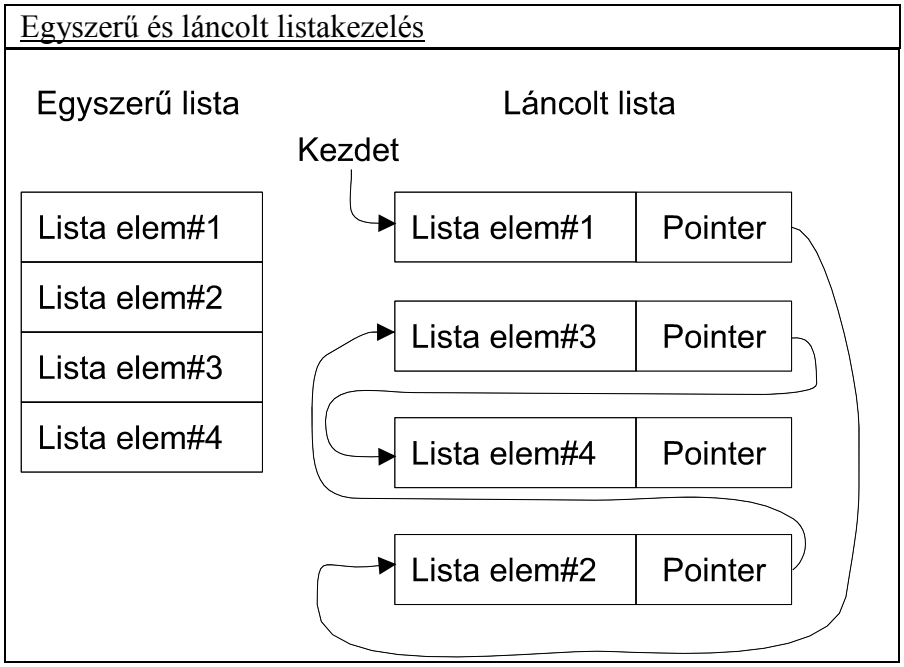
A különféle operációs rendszerek különféle adatszerkezeteket használnak a file-rendszer információk leírására. Ezen adatszerkezetek egy része fix méretű táblázat, más része ún. láncolt lista (sok kis résztáblázat, minden részben utalással, hogy hol van a folytatása). Ez utóbbit bonyolultabb karbantartani és általában lassabb benne keresni, viszont utólag változtatható a mérete.

Tipikus file-rendszer példák:

- FAT (File Allocation Table) - az MS-DOS és utódai által használt egyszerű file-rendszer. Eredetileg 200-300 kByte kapacitású floppy-lemezekhez tervezték a 80-as évek elején, emiatt a mai több Gbyte-os merevlemezeken már nem igazán hatékony. A foglaltsági információt fix táblázatban (ez maga a File Allocation Table), a könyvtár-

és file-információkat részben fix táblázatokban, részben (primitíven kialakított) láncolt listákban tárolja. Jellegzetessége a korlátozott méretű (8+3 karakteres) file-nevek használata (a Windows 95/98 ezt a korlátot egy durva beavatkozással - az eredeti FAT szabványtól eltérő többlet file-információ beiktatásával - kerüli meg).

- A nagy kapacitású merevlemezek kezelésére létrehozott mutáns verziója a FAT32, amely nagyobb méretű fix táblázatokat használ, viszont emiatt a file-rendszer információk elérése és karbantartása sokkal lassabb (és kevésbe megbízható).
- NTFS (New Technology File System) - a Windows NT által használt nagyobb teljesítményű file-rendszer. Az adatszerkezetei már majdnem mind láncolt listák, emiatt jobban megfelel nagyobb méretű merevlemezekhez. A file-nevek maximális hossza 64 karakter.



A lemezkezelés során az is lényeges szempont, hogy az egyes file-okhoz tartozó adatokat, ill. a file-rendszer leíró adatszerkezeteket hová helyezi el az operációs rendszer. A FAT és leszármazottai a file-rendszer információkat részben a lemez elejére teszik, részben szétszórják a lemezen. Az egyes file-ok írásakor pedig mindig a lemez elejéhez legközelebb eső szabad szektorokat foglalják le, emiatt egy merevlemez hosszabb használata után a sok-sok file törlése és új file-ok létrehozása, létező file-ok bővítése miatt az egyes file-ok nem egymás mellett elhelyezkedő szektorokban tárolódnak (diszk fragmentáció), emiatt az egyes file-ok írása/olvasása lelassul. Ennek kiküszöbölése érdekében a FAT file-rendszert használó merevlemezeket időnként célszerű defragmentálni, azaz az összes file-t egymás után következő szektorokba áthelyezni. Ez időigényes és kockázatos folyamat.
Más file-rendszerek (pl. az NTFS, ill. az UNIX rendszerekben használatos számos különféle file-rendszer) ügyesebben helyezik el mind a file-okat, mind pedig a file-rendszer adatstruktúrákat; pl. egy file létrehozásakor a file vége után a lemezen hagynak ki üres helyet, hogy ha később nagyobb lesz a file, akkor ezen a területen lehessen folytatni; ill. a file-rendszer információkat több csoportba osztva, a megfelelő file-okhoz minél közelebb helyezik el.

Többfeladatos, többfelhasználós operációs rendszerek esetén, ahol több független program használja egyidejűleg a merevlemezt, nem közömbös az sem, hogy az egyes lemezműveletek milyen sorrendben hajtódnak végre, ui. a lemezműveletek idejének nagy részét az író/olvasó fej mozgatása teszi ki, tehát célszerű a fejmozgást minimalizálni. Erre néhány ötlet:

- SSTF (Shortest Seek Time First) - azt a lemezműveletet végzi el legelőször, amely a fej aktuális pozíciójához legközelebb esik. Ennek a módszernek apró problémája, hogy a lemez középső területeinek elérése átlagosan gyorsabb lesz, mint a szélső területeké. (Ezt ki lehet használni arra, hogy a leggyakrabban használt adatokat - pl. file-rendszer leíró információt - eleve a lemez közepére helyezzük.)
- Pásztázás (scanning) - az író/olvasó fejet mindig csak egy irányba mozgatja, egészen addig, amíg van olyan lemezművelet, amely az aktuális pozícióhoz képest ebbe az irányba esik. Ha nincs ilyen, akkor irányt vált, és a másik irányba eső műveleteket végzi el. Változatai: N-scan (N db művelet után mindenképpen irányt vált), Circular Scan (nem vált irányt, hanem ha nincs több művelet az egyik irányban, akkor a másik irányban legmesszebbre eső műveletet hajtja végre, és ismét az eredeti irányban folytatja)

Mint arról már volt szó, a lemezműveletek sokkal lassúbbak a memóriaműveleteknél. Emiatt ha van sok szabad memória, akkor a lemezműveleteket is lehet gyorsítani ún. *disk cache* bevezetésével, azaz a memória egy részét fenn lehet tartani arra, hogy a lemez néhány egymás melletti szektorát egy mozdulattal ide bemásoljuk, és a már említett lokalitási tulajdonság miatt, amely többé-kevésbé a lemezműveletekre is vonatkozik, ezzel gyorsul a merevlemezek elérése. Figyelni kell azonban arra, hogy a disk cache-ben levő, még el nem végzett írási műveletek az operációs rendszer váratlan leállása esetén elveszhetnek, így jelentős adatvesztést, akár a file-rendszer teljes összeomlását idézve elő!

Többfelhasználós operációs rendszerek esetén az egyes file-okhoz a különböző felhasználóknak különféle hozzáférési jogaik lehetnek, amelyeket szintén a file-rendszer információk között kell eltárolni. Ennek halvány kezdeményei az MS-DOS/Windows 95 operációs rendszerekben az ún. *file-attribútumok*: csak olvasható (Read-Only), rejtett (Hidden), rendszerfile (System).
A UNIX file-rendszereiben már sokkal kifinomultabb hozzáférés-szabályozás van: minden file-nak van tulajdonosa és csoportja (a tulajdonos az a felhasználó, aki létrehozta a file-t, és minden felhasználó tagja egy vagy több csoportnak), és az olvasás, írás és programfuttatás joga külön-külön szabályozható a tulajdonos, a file-nál megjelölt csoport tagjai, valamint az összes többi felhasználó számára. Ezen jogosultságokat a file tulajdonosa szabályozhatja. Így könnyen és egyszerűen be lehet állítani, hogy a file-t azok és csak azok olvashassák, ill. írhassák, akiknek kell.
Más file-rendszerek (pl. az NTFS) *jogosultsági listák* alapján szabályozzák a file-hozzáférést: minden file-hoz tartozik egy [felhasználó, jog] elemekből álló lista, és csak olyan file-műveletek hajthatók végre, amelyek megfelelnek ennek a listának.

Egyéb perifériák kezelése

A számítógépekhez az eddig említetteken kívül még számos különféle perifériás egység csatlakoztatható. Ezek közül a nyomtatók érdemelnek leginkább figyelmet. A nyomtatók számítógéphez kapcsolásakor fellép az az alapvető probléma, hogy a nyomtatók – a működésük fizikai korlátai miatt – csak meghatározott, igen lassú tempóban képesek adatok fogadására, viszont, különösen a mai korszerű, nagy felbontású grafikus nyomtatóknál minden egyes oldal kinyomtatásához jelentős mennyiségű adatot kell elküldeni. Ezért célszerű valamilyen mechanizmust beépíteni az operációs rendszerbe, hogy a programoknak ne kelljen kivárni, amíg a nyomtatóra kikerülnek az adatok. Ezt általában az ún. spooling-gal érik el: magával a nyomtatóval egy külön kis programrész kommunikál, és a felhasználói programok nem közvetlenül a nyomtatóra, hanem egy átmeneti tárolóhelyre (spool file) küldik a nyomtatandó adatokat, ahonnan a kis program továbbítja a nyomtatóhoz. Így a

felhasználóknak nem kell kivárniuk az adatátvitelt, hanem addig is tovább dolgozhatnak.

Felhasználói felületek

Az első számítógépeket még csak meghatározott speciális célokra, az őket megépítő tudósok használatára szánták, így a kezelésük eléggé komplikált módon történt: a központi feldolgozóegység állapotát kis lámpácskák jelenítették meg, és az egyes regiszterek, valamint a memória bitjeit kapcsolókkal lehetett beállítani. A számítási eredmények leolvasása is a lámpácskák megfigyelésével történt.

Később felmerült az igény, hogy az eredményeket papíron is meg lehessen őrizni, így nyomtatásra képes egységeket kapcsoltak a számítógéphez. Ennek egyik formája az átalakított, elektromos jelekkel vezérelhető írógép volt, és így adódott az ötlet, hogy az írógép billentyűzetét adatbevitelre is felhasználják. Ez a kombináció, az ún. *konzol írógép*, széles körben elterjedt az 50-es években. Jellegzetességei az írógép-mechanizmusból adódtak: a számítógép csak egymás utáni szövegsorokat tudott kiírni, és mindent, amit a felhasználó begépelt, szintén megjelent a papíron.

Később felmerült az ötlet, hogy mivel mind a beírt adatoknak, mind a számítógép által kiírt válaszoknak nagy része nem olyan fontos, hogy papíron meg kellene örökíteni, szükség lenne olyan beviteli/megjelenítő eszközre, amely nem papírra dolgozik. Így alkották meg – a konzolírógépben a papírra nyomtató mechanizmus kicserélésével – a *képernyős terminál*t, ahol a megjelenítés már katódsugárcsöves képernyőn történik. Emiatt számos olyan lehetőséget is ad, amely a konzolírógépen nem állt rendelkezésre: a kiíratások a képernyő bármelyik pozíciójára történhetnek, nemcsak egymás alá; a megjelenítéshez használt betűtípust és színeket könnyebb módosítani, stb. A terminálok megjelenésével alakult ki a programfuttatásnak az a módja, hogy a felhasználók a programok neveit, ill. különféle parancsokat, mint szöveget beírják a billentyűzeten. Ez az ún. parancssoros felület (command line interface; az UNIX világban *shell*) – egyszerűsége és hatékonysága miatt – azóta is szinte minden operációs rendszerben megtalálható.
Körülbelül ezzel egyidőben jelentek meg a nagyobb teljesítményű nyomtatók.
Jellegzetes nagyteljesítményű eszköz a sornyomtató, korlátozott karakterkészlettel, és elég gyenge nyomtatási minőséggel.

A karakteralapú bevitel és megjelenítés azonban számos feladatra (pl. grafika) nem alkalmas. A 60-as évek végén jelentek meg az első kísérleti grafikus felhasználói felületek (GUI: Graphical User Interface). Ezeknél a bevitelt a billentyűzet mellett egy szabadon mozgatható pozicionáló eszköz (tipikusan egér) segíti; az egyes programok az adataik megjelenítését különálló ablakokban végzik, amelyek egymástól függetlenül mozgathatók, átméretezhetők stb., a programok indítása és a felhasználók műveleteinek végrehajtása pedig különféle grafikai elemek (ikonok, legördülő menük, nyomógombok, tolókák stb.) aktiválásával végezhető. A ma elterjedt operációs rendszerek legtöbbje – kényelmessége és hatékonysága miatt – valamilyen grafikus felületre épül, bár a parancssoros kezelési lehetőség általában szintén megtalálható bennük.

A Windows sorozat eleinte az MS-DOS grafikus kiegészítése volt (Windows 3.1), innen fejlődött többé-kevésbé önálló operációs rendszerré, amíg mára a világszerte legelterjedtebb grafikus felület lett.

A UNIX-ok grafikus felülete, az X11 (avagy X Windows), jóval korábbról származik (tkp. az összes többi grafikus felülethez az X11-ből és rokonaiból "kölcsönözték" a többi gyártók az

ötleteket), és moduláris felépítésének köszönhetően sokkal inkább testre szabható, sokkal rugalmasabban illeszthető a felhasználó igényeihez az egyes komponensek lecserélésével; ennek persze az az ára, hogy a felhasználótól több munkát és nagyobb hozzáértést kíván, és a különféle X11-variációk megjelenése és kezelése messze nem annyira egységes, mint a Windows-é.

A Windows története

Release date	Product name	Current Version	Notes	Last IE
November 1985	Windows 1.01	1.01	Unsupported	-
November 1987	Windows 2.03	2.03	Unsupported	-
May 1988	Windows 2.10	2.10	Unsupported	-
March 1989	Windows 2.11	2.11	Unsupported	-
May 1990	Windows 3.0	3.0	Unsupported	-
March 1992	Windows 3.1x	3.1	Unsupported	5
October 1992	Windows For Workgroups 3.1	3.1	Unsupported	5
July 1993	Windows NT 3.1	NT 3.1	Unsupported	5
December 1993	Windows For Workgroups 3.11	3.11	Unsupported	5
January 1994	Windows 3.2 (released in Simplified Chinese only)	3.2	Unsupported	5
September 1994	Windows NT 3.5	NT 3.5	Unsupported	5
May 1995	Windows NT 3.51	NT 3.51	Unsupported	5
August 1995	Windows 95	4.0.950	Unsupported	5.5
July 1996	Windows NT 4.0	NT 4.0.1381	Unsupported	6
June 1998	Windows 98	4.10.1998	Unsupported	6
May 1999	Windows 98 SE	4.10.2222	Unsupported	6
February 2000	Windows 2000	NT 5.0.2195	Extended Support until July 13, 2010[22]	6
September 2000	Windows Me	4.90.3000	Unsupported	6
October 2001	Windows XP	NT 5.1.2600	Extended Support until July 13, 2010 for SP2 and April 8, 2014 for SP3. (RTM and SP1 unsupported).	8
March 2003	Windows XP 64-bit Edition (IA-64)	NT 5.2.3790	Unsupported	6
April 2003	Windows Server 2003	NT 5.2.3790	Current (RTM unsupported).	8
April 2005	Windows XP Professional x64 Edition	NT 5.2.3790	Extended Support until July 13, 2010 for SP2 and April 8, 2014 for SP3. (RTM and SP1 unsupported).	8
July 2006	Windows Fundamentals for Legacy PCs	NT 5.1.2600	Current	8
November 2006 (volume licensing) January 2007 (retail)	Windows Vista	NT 6.0.6002	Current except for RTM which will be unsupported as of April 13, 2010 Version changed to NT 6.0.6001 with SP1 (February 4, 2008) and to NT 6.0.6002 with SP2 (April 28, 2009).	9
July 2007	Windows Home Server	NT 5.2.4500	Current	8
February 2008	Windows Server 2008	NT 6.0.6002	Current Version changed to NT 6.0.6002 with SP2 (April 28, 2009).	9
October 2009	Windows 7 and Windows Server 2008 R2	NT 6.1.7600	Current	9
2012	Windows 8	Unknown	Upcoming	Unknown