**Operációs rendszerek fogalma, felépítése, osztályozásuk**

Az ISO nemzetközi szabványosítási szervezet definíciója szerint az operációs rendszer: "Olyan programrendszer, amely a számítógépes rendszerben a programok végrehajtását vezérli: így például ütemezi a programok végrehajtását, elosztja az erőforrásokat, biztosítja a felhasználó és a számítógépes rendszer közötti kommunikációt."

Az operációs rendszerek leggyakoribb csoportosítási módjai a következőek:

1. Felhasználók szempontjából: egyfelhasználós (DOS), többfelhasználós (UNIX, Windows)
2. Egyidőben futtatható programok száma: egyfeladatos (DOS), többfeladatos
3. Megvalósítás szerint: interaktív, nem interaktív
4. A gép méretétől függő: mikrogépes, kisgépes, nagygépes (mainframe)
5. Terjesztési licenc alapján: kereskedelmi (DOS, Win, néhány UNIX), szabad (UNIX)

Az operációs rendszer komponensei:

* rendszermag (kernel): Feladata a hardver optimális kihasználásának irányítása, a kért programok futtatása, alkalmazói kéréseknek kiszolgálása.
* API (Application Programing Interface): Olyan szabálygyűjtemény, mely leírja hogyan kell kérni a szolgáltatásokat a kerneltől, és a kernel válaszát hogyan kapjuk meg.
* Rendszerhéj (shell): Feladata a parancsértelmezés. Lehet a shell parancssoros ( CLI - Command Line Interface - mint, pl. DOS), vagy grafikus - GUI - felületű (pl. Windows)
* Szerviz- és segédprogramok (utility): a felhasználói "élményt" fokozó kiegészítő programok (pl szövegszerkesztők, fordítóprogramok), amelyek nem képzik a rendszer elválaszthatatlan részét.

**Fájlok és fájlrendszerek**

A számítógépen lévő információtárolási egysége a **fájl** (file). Egy fájl tartalma a gép szempontjából vagy adat, vagy program.

A fájlok alapvető jellemzői:

* fájlnév
* kiterjesztés
* méret
* dátum
* idő

A **fájlrendszer** szűkebb értelemben nem más, mint a fájlok tárolására és rendszerezésére kialakított struktúra egy háttértároló eszközön (pl. floppy lemezen vagy merevlemezen vagy CD-ROM-on), vagy annak egy részén (pl.: merevlemez egy partícióján).  
A fájlrendszer UNIX esetén egy tiszta fogalom, mert egyetlenegy gyökér (root - / ) könyvtárral rendelkezik, míg a DOS és Windows estén nem, mert ott meghajtónként van egy-egy.

Precízebben meghatározva: *egy* ***fájlrendszer*** *absztrakt adattípusok halmaza, amelyeket adatok tárolására, hierarchikus rendezésére, kezelésére, megtalálására illetve navigálásra, hozzáférésre, és visszakeresésére valósítottak meg*.

A legismertebb fájlrendszerek egy adattároló eszköz segítségével biztosítják, hogy elérhető a terület elérhető legyen egy fix méretű blokkokból álló sorozat képében (ahol az általában 512 byte méretű blokkokat gyakran szektoroknak is nevezik, de ez a méret lehet akár 1,2 vagy 4 kilobyte is). A fájlrendszer feladata, hogy ezeket a szektorokat fájlokká valamint katalógusokká szervezze össze, és tartsa nyilván, melyik szektor melyik fájlhoz tartozik, és melyik szektorok nem használhatók már tárolásra. A legtöbb fájlrendszer ezen felül bevezeti még a klaszter (cluster) fogalmát is, amely a lemezről lefoglalható legkisebb tárterület méretét határozza meg, ez többnyire több blokkból áll és mérete formázáskor dől el.

Ami mindenképp létezik, az a fájlnevek és a fájlok összekapcsolását végző **metaadatok**, ami többnyire valamilyen index, pl. a DOS esetén a FAT táblázatban, míg Unix vonalon az i-node számoknál.  
Hierarchiába történő szervezés == mappák

**Speciális fájlok Unix alatt**

* **link**: hivatkozás más fájlra
* **nevesített csővezeték** (named pipe): folyamatok közötti kommunikációra ad lehetőséget, oly módon hogy az egyik alkalmazás kimenetét egy másik alkalmazás bemenetére köti. Egyszerű FIFO pufferekről van szó, amikbe írni és amikből olvasni lehet
* **socket**: speciális fájl melyet a pipeokhoz hasonlóan, folyamatok közötti kommunikációra használnak, de immár hálózatos környezetbe szerver-kliens kommunikáció során.
* **eszköz** (device) fájlok: ezek a hardver elemeit reprezentálják a fájlrendszerben így hozzáférési jogokat adhatunk az egyes elemeknek, és utasításokban használhatjuk közvetlenül magát az eszközöket.

**Átirányítás, csővezetékek**

Átirányítás

A szabványos bemenet (stdin), a szabványos kimenet (stdout) és a szabványos hibakimenet (stderr) átirányítása. Mind a bemenet, mind pedig a kimenet (hibakimenet) átirányítható egy tetszőleges állományba: <, >, >>, 2>, &>

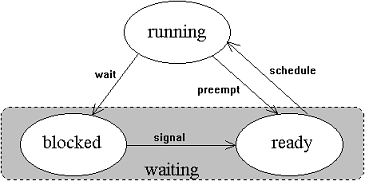
Csővezetékek

A cső vagy csővezeték (pipe, pipeline) a programok egy olyan sorozata, amelyek a szabványos folyamaik által vannak összekötve, azaz a Program1 kimenetét (stdout) a Program2 bemenetére (stdin) köti.

**Folyamatkezelés**

A Linux többfeladatos (multitask) és többfelhasználós (multiuser) rendszer. Ebből következik, hogy akár egy felhasználó egy időben több programot is futtathat. Az elindított program a processz, azaz folyamat, más megfogalmazásban egy végrehajtható fájl "élő" változata. Gyakran taszknak is nevezik. A folyamatok jól definiált hierarchiát alkotnak. Minden folyamatnak pontosan egy szülője (parent) van, és egy vagy több gyermek folyamata (child process) lehet. A folyamat hierarchia tetején az init folyamat helyezkedik el.  
A Linux minden egyes feladathoz két számot (PID, process identificator - feladat azonosítót és a PPID, parent process identification - szülő azonosítója) rendel. A rendszer a PID-et automatikusan növeli. Az init folyamat PID-je 1.

A legegyszerűbb és legáltalánosabb állapot és állapotátmenet diagram a következő:



Folyamatkezelő parancsok:

* ps: A Process Sate parancs processzusok állapotát jeleníti meg.

Szintaktika:

ps [kapcsolók]

* pstree: Az initből induló folyamathierarchiát lehet a parancs segítségével megtekintetni fa szerkezetű ábrázolásban.
* nohup: Mikor a rendszerből kijelentkezünk (azaz a bash bezáródik) minden gyerekfolyamatát a rendszer automatikusan kilövi. Lehetőségünk van azonban arra is, hogy egy folyamatot immúnissá tegyünk kilépésünkre. Hosszan, több óráig, több napig futó programokat a nohup paranccsal indíthatunk.
* top: A top a kill és a ps parancs egyesített változata mely folyamatosan futva listázza az éppen aktív folyamatokat, információt nyújt a rendszer állapotáról és terheltségi mutatóiról illetve lehetőséget ad szignálok küldésére folyamatok számára.

**Jelzések, szignálok**

A Linux rendszer a folyamatok vezérlését a folyamatoknak küldött ún. szignálok segítségével végzi: a Ctrl+Z billentyű például egy STOP szignált küld az előtérben futó processznek.

* kill: A kill a nevével ellentétben nem csak folyamatok megölésére használható, hanem tetszőlegese signalt küldhetünk vele bármely folyamatnak melynek tudjuk a tudjuk a PID számát és rendelkezünk a folyamat kezeléséhez megfelelő jogokkal. Alapértelmezés szerint (signal paraméter nélkül használva) a kill egy TERM (terminate) szignált küld a megadott folyamatnak.
* HUP, TERM, ALARM, KILL

**Ütemezett végrehajtás**

A UNIX multitaszking képessége nem korlátozódik csak a jelenben a futó folyamatokra. Tartalmaz programokat, amelyek lehetővé teszik a programok ütemezett futtatását, akár egyszeri, akár ismétlődő időközönként. Egy adott időben az at program segítségével indíthatunk el folyamatokat, ismétlődő esetekben pedig a crontab alkalmazással.

at szintaxisa:

$at 10:30am today

$at midnight

$at 12:01 1 jan 2012

at> végrehajtandó parancs